**Please turn your camera and microphone OFF !**

# RIKEN International HPC Summer School 2024 -Toward Society 5.0- (IHPCSS)
# Overview of the Class

https://www.r-ccs.riken.jp/en/outreach/schools/20240902-0910/

Kengo Nakajima
RIKEN Center for Computational Science (R-CCS)

- Target: Parallel FEM
- Supercomputers and Computational Science
- Overview of the Class
- Future Issues

This 5-day intensive course provides introduction to large-scale scientific computing using the most advanced massively parallel supercomputers. Topics cover:

- Parallel Finite-Element Method (FEM)
  - Finite-Element Method (FEM)
  - Message Passing Interface (MPI)
  - Parallel FEM using MPI and OpenMP
  - Parallel Numerical Algorithms for Iterative Linear Solvers
  - Parallel Numerical Libraries (e.g. PETSc)
- Introductions to AI/DNN (Deep Neural Network)

Several sample programs will be provided and participants can review the contents of lectures through hands-on-exercise/practices using **Wisteria/BDEC-01 (Odyssey) with A64FX Processors (Fugaku Compatible System)** at the University of Tokyo.

Finite-Element Method is widely-used for solving various types of real-world scientific and engineering problems, such as structural analysis, fluid dynamics, electromagnetics, and etc. This lecture course provides brief introduction to procedures of FEM for 1D/3D steady-state heat conduction problems with iterative linear solvers and to parallel FEM. **Lectures for parallel FEM will be focused on design of data structure for distributed local mesh files, which is the key issue for efficient parallel FEM.** Introduction to MPI (Message Passing Interface), which is widely used method as "de facto standard" of parallel programming, is also provided.

Solving large-scale linear equations with sparse coefficient matrices is the most expensive and important part of FEM and other methods for scientific computing, such as Finite-Difference Method (FDM) and Finite-Volume Method (FVM). Recently, families of Krylov iterative solvers are widely used for this process. In this course, details of implementations of parallel Krylov iterative methods are provided along with parallel FEM.

Moreover, lectures on programming for multicore architectures will be also given along with brief introduction to OpenMP and OpenMP/MPI Hybrid Parallel Programming Model.

Finally, lectures and hands-on for using PETSc library will be provide in the morning of the 5$^{th}$ day.

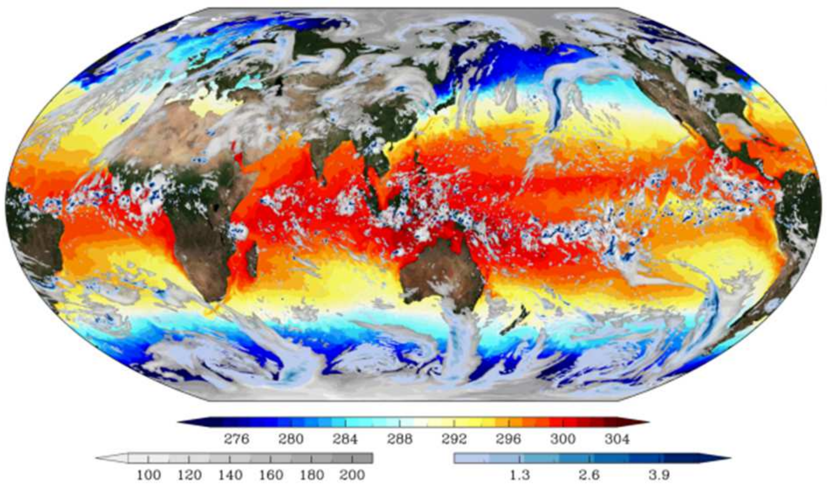
Rapid progress has been made in recent years in artificial intelligence technology, supported by the evolution of the Deep Neural Network (DNN) and the growing use of AI-specific hardware to support this technology. Understanding the basic mathematical background of DNNs, including the concept of Neural Networks as the basis of DNN, their capabilities as high-performance nonlinear approximation functions, and how to determine themselves efficiently, is not necessarily essential for AI users but is regarded as significant for researchers who are further advancing the field.

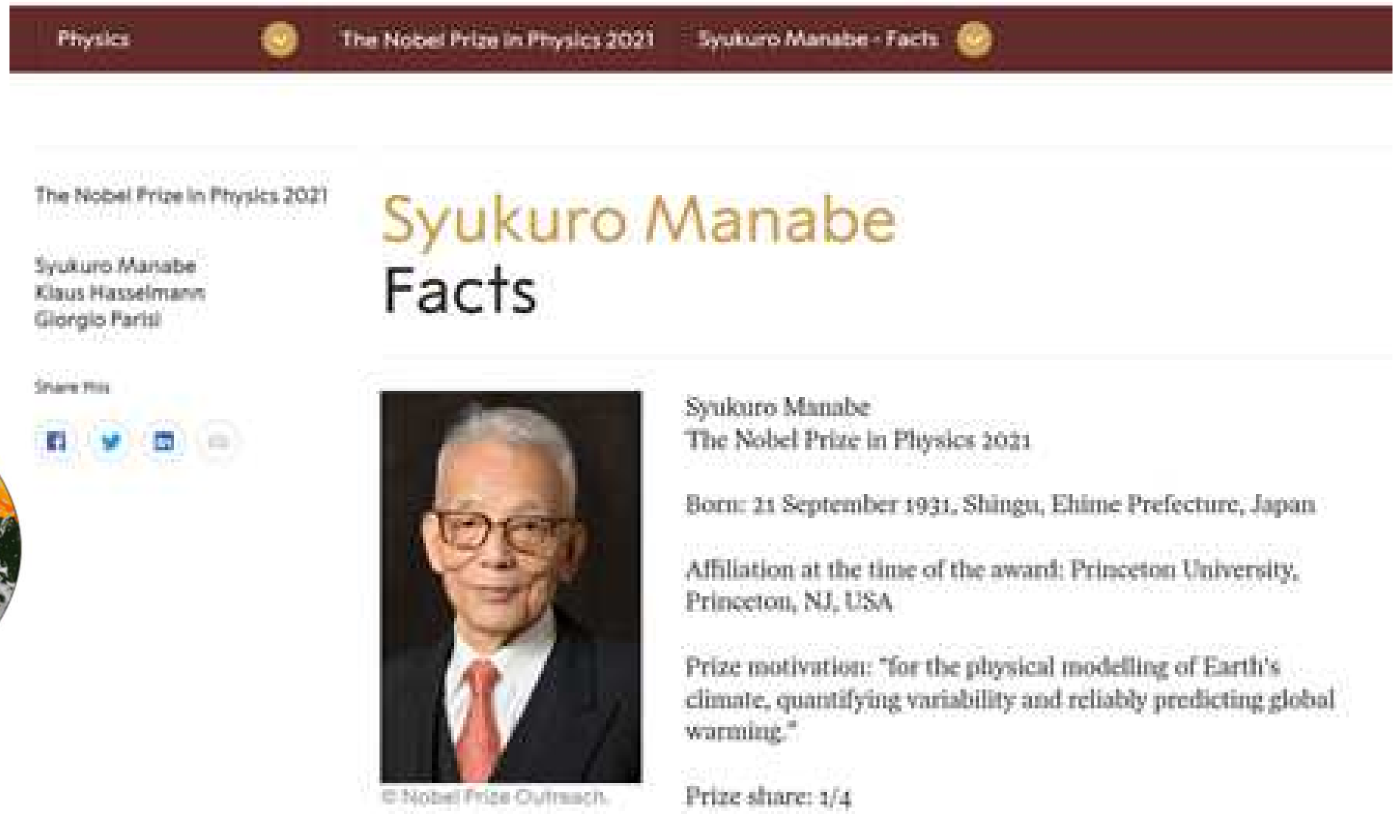
In this seminar, we will start with the basics of Neural Networks in a short lecture and then move on to approximating functions by backpropagation (i.e., the learning process in artificial intelligence), which is the underlying technology for the current DNNs. In addition, the seminar aims to deepen understanding through practical exercises such as image categorization using a simple program and the standard MNIST image database, identification of hand-written weird kana characters (namely Hentai-Gana;「変体仮名」in Japanese), and bridge to the use of higher-level AI frameworks and toolkits.

The following part of this material mainly focuses on the Parallel FEM in first 4-days

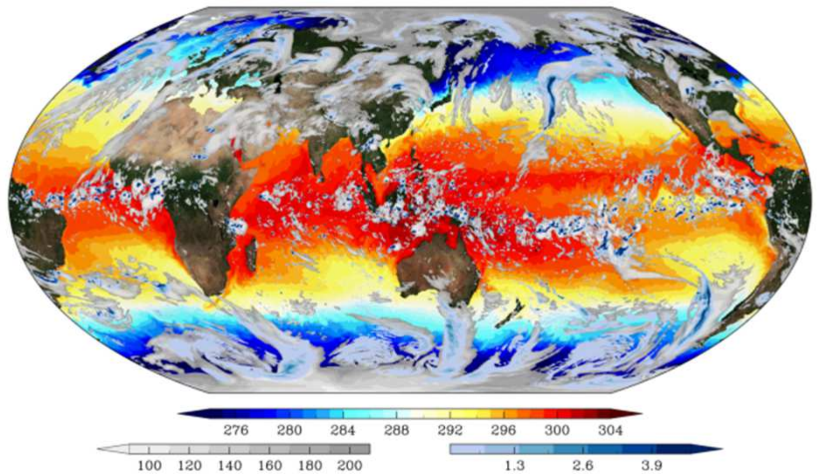
# Motivation for Parallel Computing (and this class)

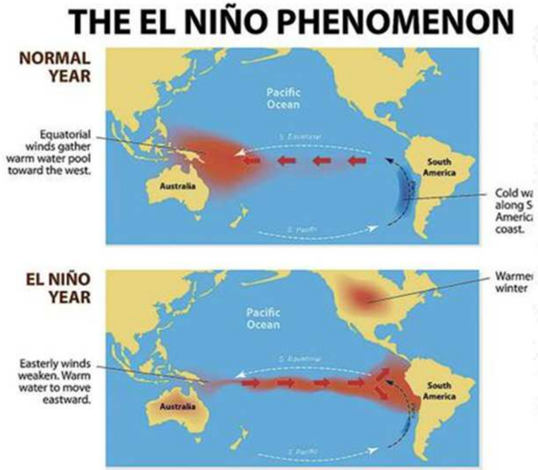
- Large-scale parallel computer enables fast computing in large-scale scientific simulations with detailed models. Computational science develops new frontiers of science and engineering.

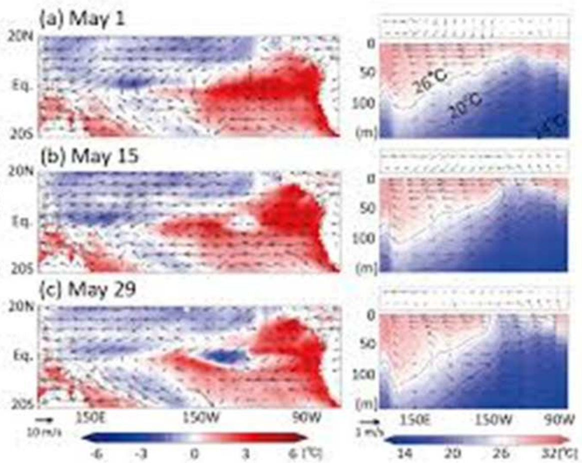


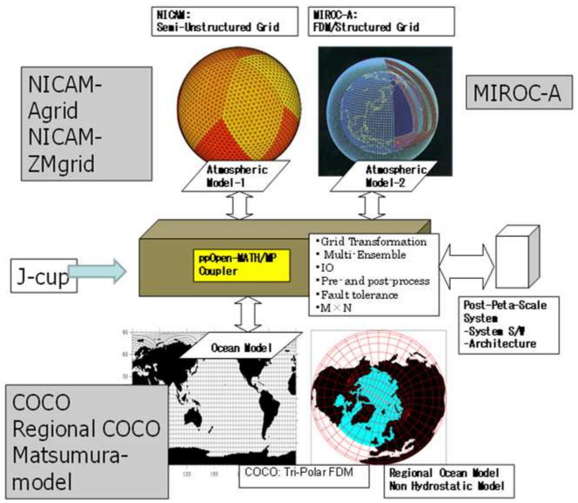



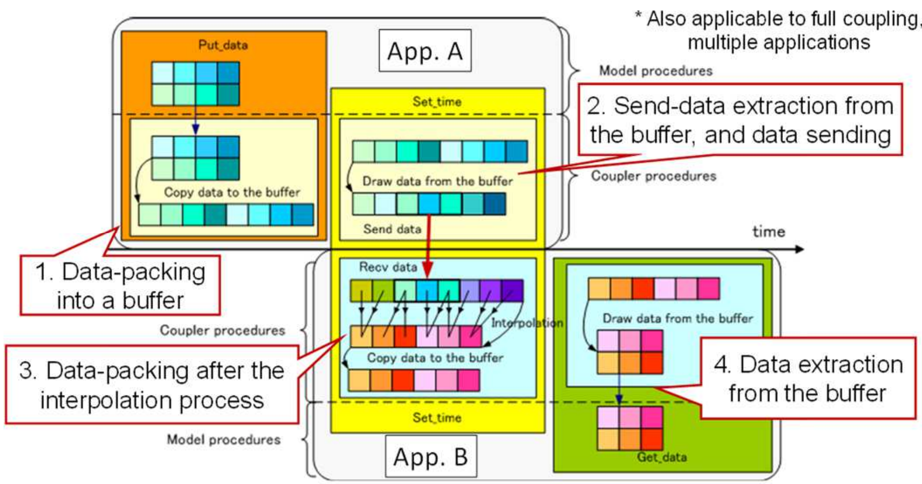
# Global Atmosphere-Ocean Coupled Simulations

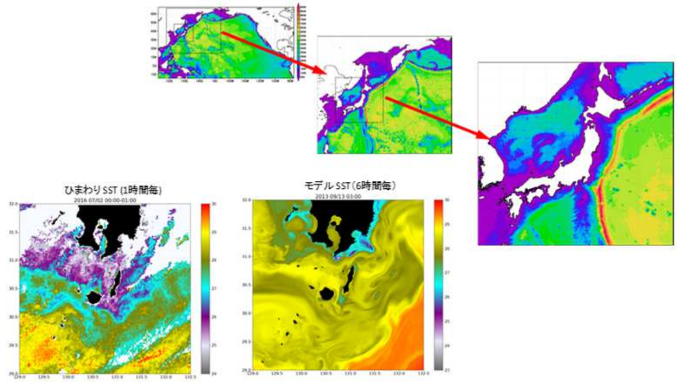



[c/o Prof. M. Sato, Prof. H. Hasumi (AORI/U.Tokyo)]

# Motivation for Parallel Computing (and this class)

- Large-scale parallel computer enables fast computing in large-scale scientific simulations with detailed models. Computational science develops new frontiers of science and engineering.
- Why parallel computing ?
  - faster & larger
  - "larger" is more important from the view point of "new frontiers of science & engineering", but "faster" is also important.
  - + more complicated
  - Ideal: Scalable
    - Weak Scaling, Strong Scaling

# Scalable, Scaling, Scalability

- Solving $N^x$ scale problem using $N^x$ computational resources during same computation time
  - for large-scale problems: **Weak Scaling, Weak Scalability**
  - e.g. CG solver: more iterations needed for larger problems
- Solving a problem using $N^x$ computational resources during 1/N computation time
  - for faster computation: **Strong Scaling, Strong Scalability**

# Scientific Computing = SMASH

- **<u>S</u>cience**
- **<u>M</u>odeling**
- **<u>A</u>lgorithm**
- **<u>S</u>oftware**
- **<u>H</u>ardware**

- You have to learn many things.
- Collaboration (or Co-Design) will be important for future career of each of you, as a scientist and/or an engineer.
  - You have to communicate with people with different backgrounds.
  - It is more difficult than communicating with foreign scientists from same area.
- (Q): Your Department ?

# This Class ...

**Science**

**Modeling**

**Algorithm**

**Software**

**Hardware**

- Parallel FEM using MPI and OpenMP

- Science: Heat Conduction
- Modeling: FEM
- Algorithm: Iterative Solvers etc.

- You have to know many components to learn FEM, although you have already learned each of these in undergraduate and high-school classes.

# Road to Programming for “Parallel” Scientific Computing

**Programming for Parallel Scientific Computing (e.g. Parallel FEM/FDM)**

**Programming for Real World Scientific Computing (e.g. FEM, FDM)**

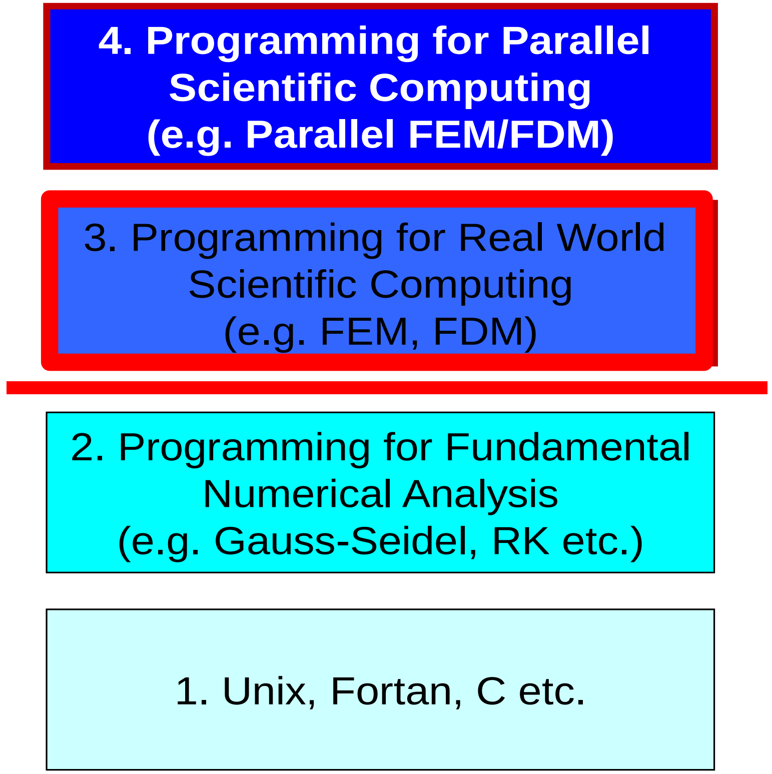
**Big gap here !!**

Programming for Fundamental Numerical Analysis (e.g. Gauss-Seidel, RK etc.)

Unix, Fortan, C etc.

# The third step is important !

- How to parallelize applications ?
  - How to extract parallelism ?
  - If you understand methods, algorithms, and implementations of the original code, it's easy.
  - "Data-structure" is important



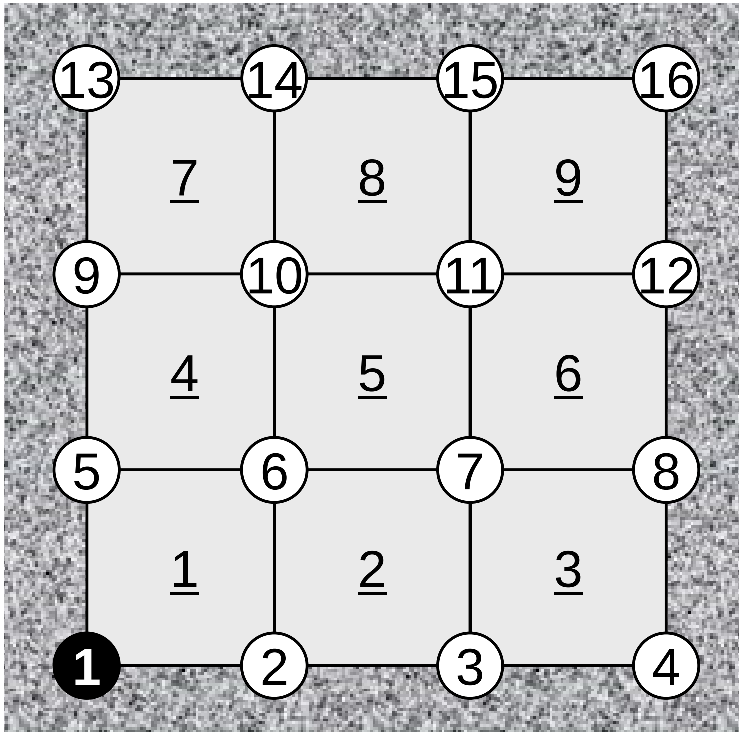

- How to understand the code ?
  - Reading the application code !!
  - It seems primitive, but very effective.
  - In this class, "reading the source code" is encouraged.

# Finite-Element Method (FEM)

- One of the most popular numerical methods for solving PDE's.
  - elements (meshes) & nodes (vertices)
- Consider the following 2D heat transfer problem:

$$\lambda\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}\right)+Q=0$$

  - 16 nodes, 9 bi-linear elements
  - uniform thermal conductivity (λ=1)
  - uniform volume heat flux (Q=1)
  - T=0 at node 1
  - Insulated boundaries



# Galerkin FEM procedures

- Apply Galerkin procedures to each element:

$$\int_V [N]^T \left\{ \lambda \left( \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) + Q \right\} dV = 0$$

where $T = [N]\{\phi\}$ in each elem.

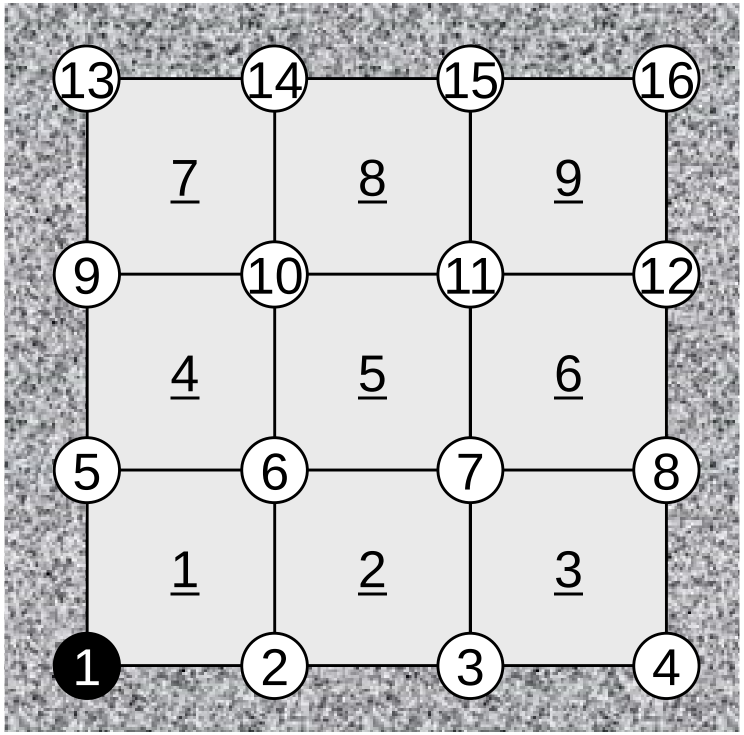
$\{\phi\}$ : $T$ at each vertex

$[N]$ : Shape function (Interpolation function)

- Introduce the following "weak form" of original PDE using Green's theorem:

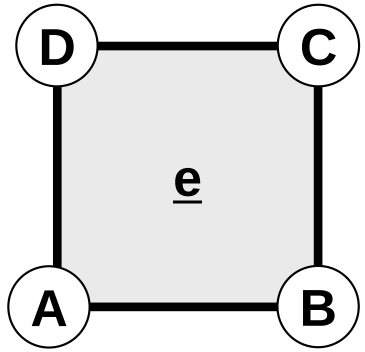
$$-\int_V \lambda \left( \frac{\partial [N]^T}{\partial x} \frac{\partial [N]}{\partial x} + \frac{\partial [N]^T}{\partial y} \frac{\partial [N]}{\partial y} \right) dV \cdot \{\phi\}$$

$$+ \int_V Q [N]^T dV = 0$$



# Element Matrix

- Apply the integration to each element and form "element" matrix.




$$-\int_V \lambda \left( \frac{\partial [N]^T}{\partial x} \frac{\partial [N]}{\partial x} + \frac{\partial [N]^T}{\partial y} \frac{\partial [N]}{\partial y} \right) dV \cdot \{\phi\}$$

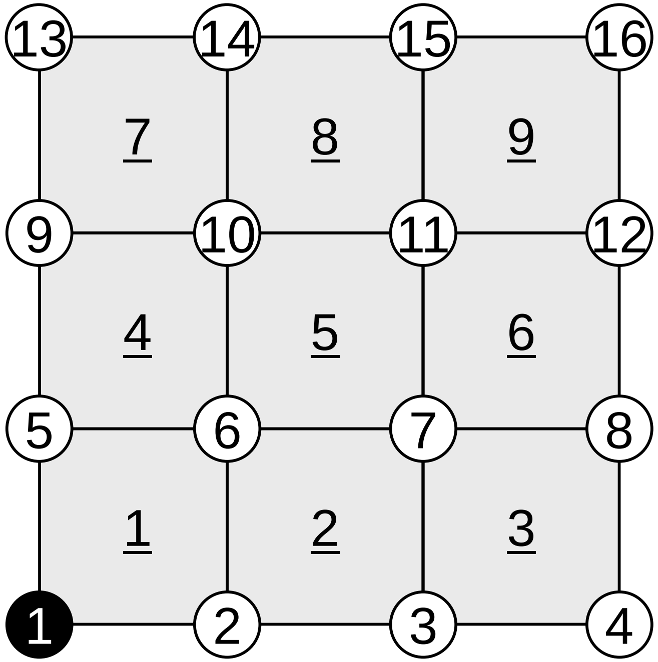
$$+\int_V Q [N]^T dV = 0$$

$$[k^{(e)}]\{\phi^{(e)}\} = \{f^{(e)}\}$$

$$\begin{bmatrix} k_{AA}^{(e)} & k_{AB}^{(e)} & k_{AC}^{(e)} & k_{AD}^{(e)} \\ k_{BA}^{(e)} & k_{BB}^{(e)} & k_{BC}^{(e)} & k_{BD}^{(e)} \\ k_{CA}^{(e)} & k_{CB}^{(e)} & k_{CC}^{(e)} & k_{CD}^{(e)} \\ k_{DA}^{(e)} & k_{DB}^{(e)} & k_{DC}^{(e)} & k_{DD}^{(e)} \end{bmatrix} \begin{Bmatrix} \phi_A^{(e)} \\ \phi_B^{(e)} \\ \phi_C^{(e)} \\ \phi_D^{(e)} \end{Bmatrix} = \begin{Bmatrix} f_A^{(e)} \\ f_B^{(e)} \\ f_C^{(e)} \\ f_D^{(e)} \end{Bmatrix}$$

# Global (Overall) Matrix

Accumulate each element matrix to “global” matrix.

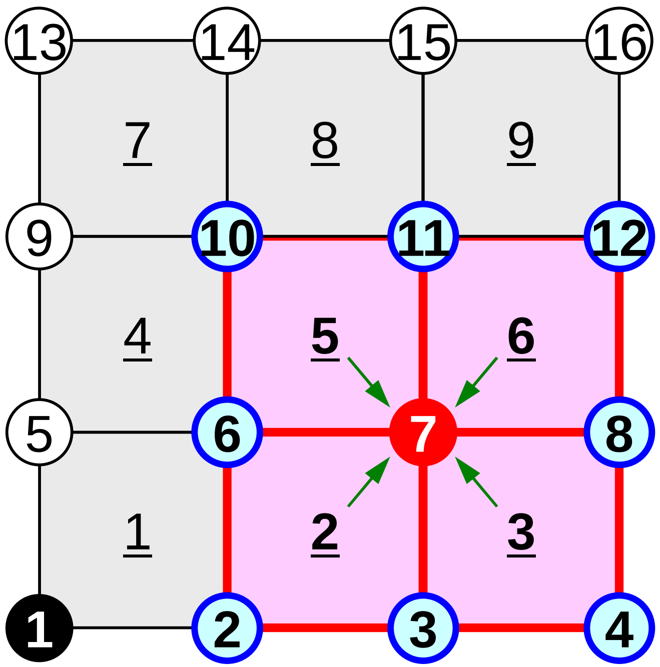



$$[K]\{\Phi\}=\{F\}$$

$$\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
& X & D & X & & X & X & X & & & & & & & & \\
& & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
& X & X & X & & X & D & X & & X & X & X & & & & \\
& & X & X & & & X & D & & & X & X & & & & \\
& & & & X & X & & & D & X & & & X & X & & \\
& & & & X & X & X & & X & D & X & & X & X & X & \\
& & & & & X & X & X & & X & D & X & & X & X & X \\
& & & & & & X & X & & & X & D & & & X & X \\
& & & & & & & & X & X & & & D & X & & \\
& & & & & & & & X & X & X & & X & D & X & \\
& & & & & & & & & X & X & X & & X & D & X \\
& & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix}
\Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \\ \Phi_{10} \\ \Phi_{11} \\ \Phi_{12} \\ \Phi_{13} \\ \Phi_{14} \\ \Phi_{15} \\ \Phi_{16}
\end{Bmatrix}
=
\begin{Bmatrix}
F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \\ F_{10} \\ F_{11} \\ F_{12} \\ F_{13} \\ F_{14} \\ F_{15} \\ F_{16}
\end{Bmatrix}$$

# To each node …

Effect of surrounding elem's/nodes are accumulated.




$$[K]\{\Phi\}=\{F\}$$

$$
\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
& X & D & X & & X & X & X & & & & & & & & \\
& & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
& X & X & X & & X & D & X & & X & X & X & & & & \\
& & X & X & & & X & D & & & X & X & & & & \\
& & & & X & X & & & D & X & & & X & X & & \\
& & & & X & X & X & & X & D & X & & X & X & X & \\
& & & & & X & X & X & & X & D & X & & X & X & X \\
& & & & & & X & X & & & X & D & & & X & X \\
& & & & & & & & X & X & & & D & X & & \\
& & & & & & & & X & X & X & & X & D & X & \\
& & & & & & & & & X & X & X & & X & D & X \\
& & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix}
\Phi_1\\ \Phi_2\\ \Phi_3\\ \Phi_4\\ \Phi_5\\ \Phi_6\\ \Phi_7\\ \Phi_8\\ \Phi_9\\ \Phi_{10}\\ \Phi_{11}\\ \Phi_{12}\\ \Phi_{13}\\ \Phi_{14}\\ \Phi_{15}\\ \Phi_{16}
\end{Bmatrix}
=
\begin{Bmatrix}
F_1\\ F_2\\ F_3\\ F_4\\ F_5\\ F_6\\ F_7\\ F_8\\ F_9\\ F_{10}\\ F_{11}\\ F_{12}\\ F_{13}\\ F_{14}\\ F_{15}\\ F_{16}
\end{Bmatrix}
$$

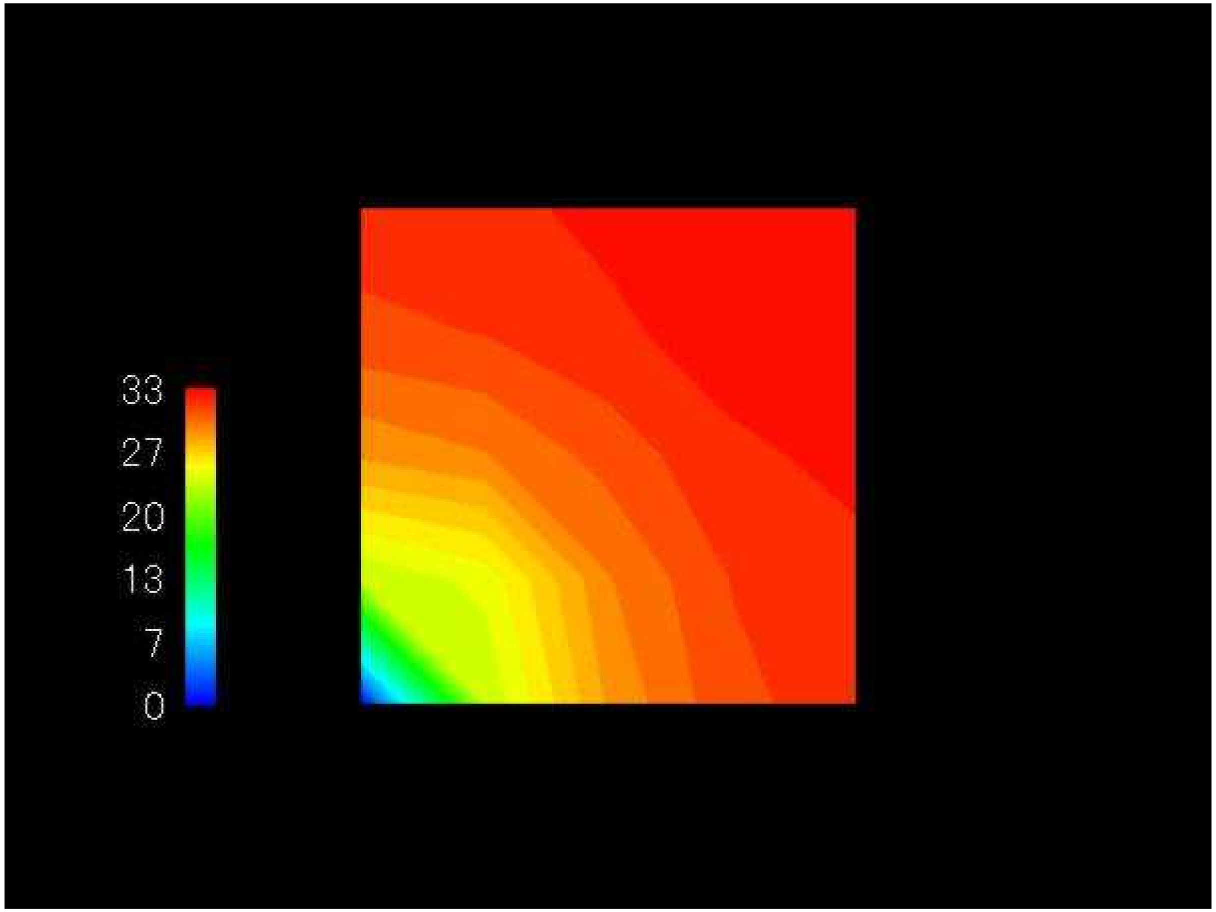
**Coefficient matrix is very "sparse", many "0's"**
**Contributions from only neighbors**

# Solve the obtained global eqn’s

under certain boundary conditions
($\Phi_1$=0 in this case)

$$
\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
& X & D & X & & X & X & X & & & & & & & & \\
& & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
& X & X & X & & X & D & X & & X & X & X & & & & \\
& & X & X & & & X & D & & & X & X & & & & \\
& & & & X & X & & & D & X & & & X & X & & \\
& & & & X & X & X & & X & D & X & & X & X & X & \\
& & & & & X & X & X & & X & D & X & & X & X & X \\
& & & & & & X & X & & & X & D & & & X & X \\
& & & & & & & & X & X & & & D & X & & \\
& & & & & & & & X & X & X & & X & D & X & \\
& & & & & & & & & X & X & X & & X & D & X \\
& & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix}
\Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \\ \Phi_{10} \\ \Phi_{11} \\ \Phi_{12} \\ \Phi_{13} \\ \Phi_{14} \\ \Phi_{15} \\ \Phi_{16}
\end{Bmatrix}
=
\begin{Bmatrix}
F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \\ F_{10} \\ F_{11} \\ F_{12} \\ F_{13} \\ F_{14} \\ F_{15} \\ F_{16}
\end{Bmatrix}
$$

# Result …

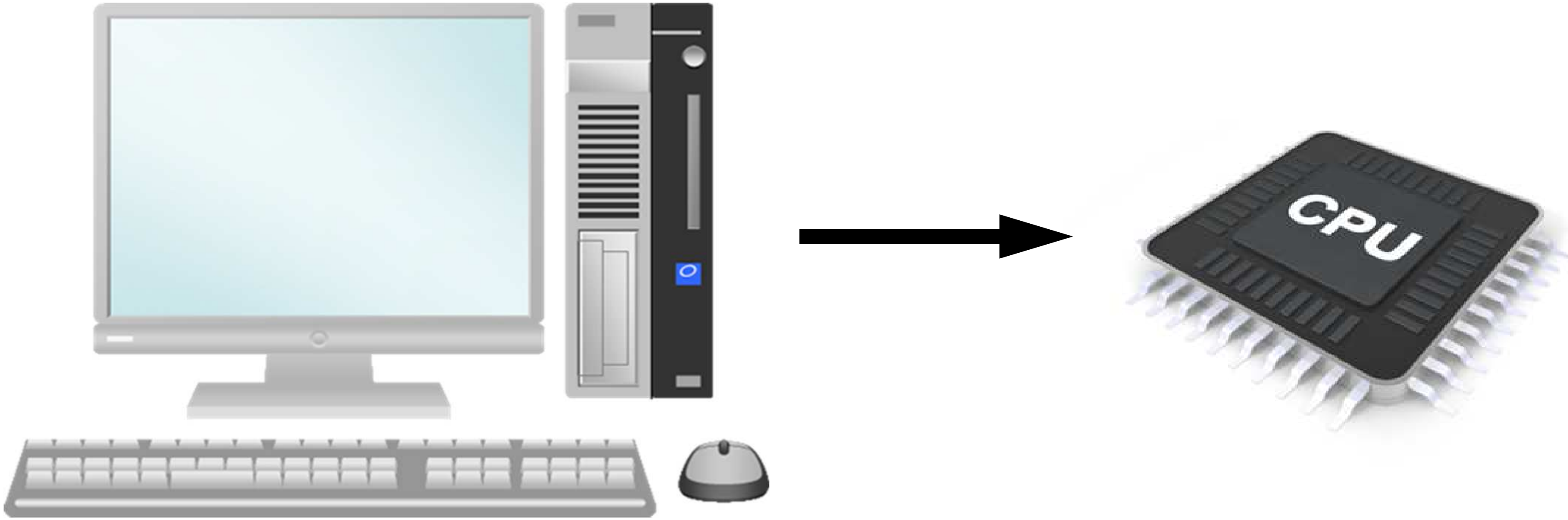


# Features of FEM applications

- Typical Procedures for FEM Computations
  - Input/Output
  - Matrix Assembling
  - Linear Solvers for Large-scale Sparse Matrices
  - Most of the computation time is spent for matrix assembling/formation and solving linear equations.
- **HUGE** “indirect” accesses
  - memory intensive
- Local “element-by-element” operations
  - sparse coefficient matrices
  - suitable for parallel computing
- Excellent modularity of each procedure

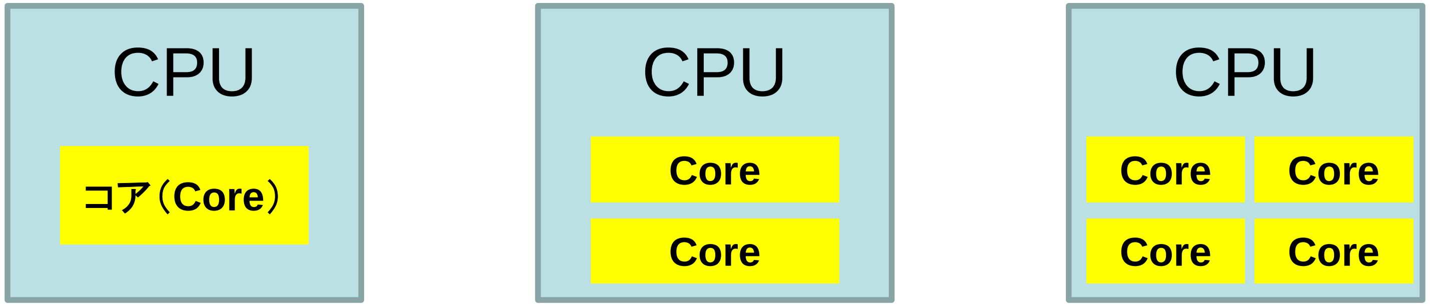
- Target: Parallel FEM
- **Supercomputers and Computational Science**
- Overview of the Class
- Future Issues

# Computer & CPU




- Central Processing Unit（中央処理装置）: CPU
- CPU's used in PC and Supercomputers are based on same architecture
- GHz: Clock Rate
  - Frequency: Number of operations by CPU per second
    - GHz -> $10^9$ operations/sec

# Multicore CPU




Single Core
1 cores/CPU

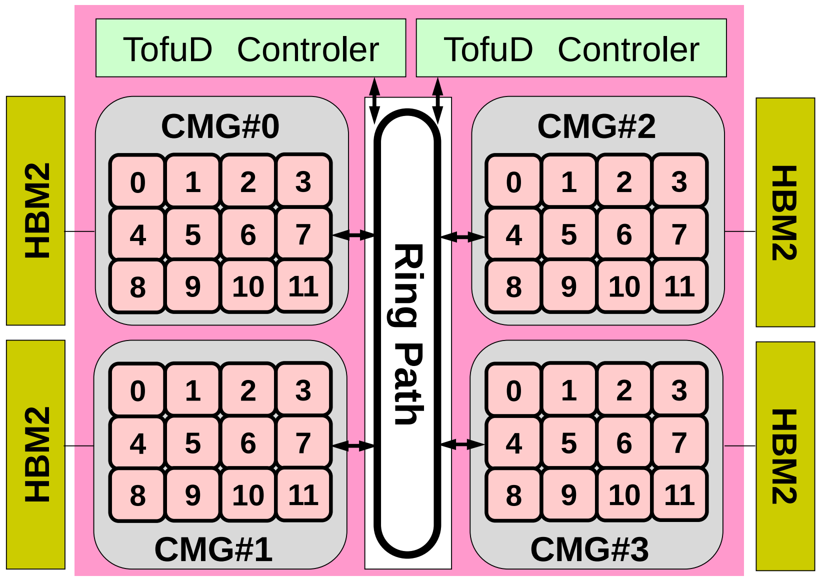
Dual Core
2 cores/CPU

Quad Core
4 cores/CPU

- Core= Central part of CPU
- Multicore CPU's with 4-8 cores are popular
  - Low Power

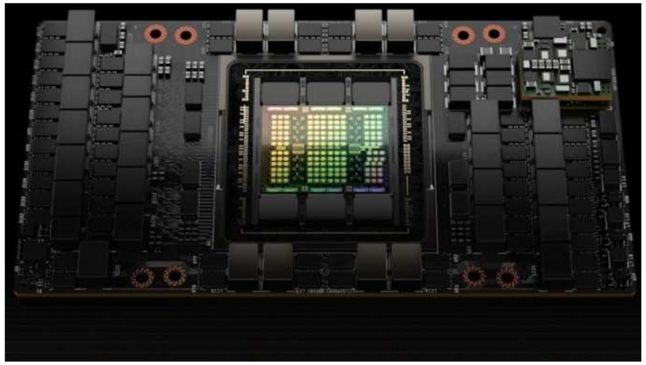
- GPU: Manycore
  - $O(10^1)$-$O(10^2)$ cores
- More and more cores
  - Parallel computing

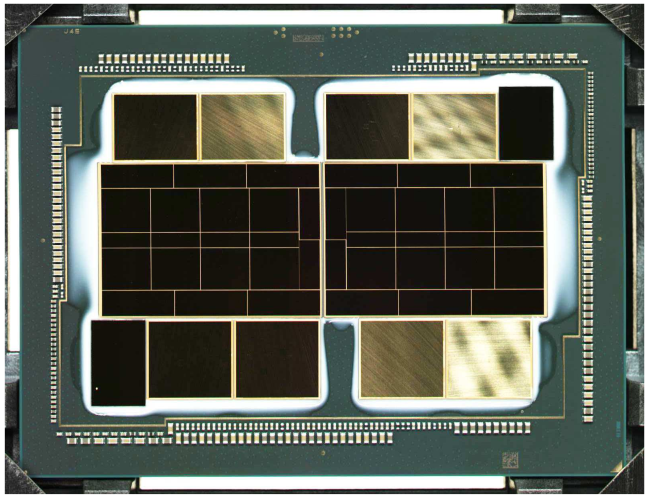
- **Odyssey: 48-cores/node**
  - **Fujitsu/ARM A64FX**



# GPU/Accelerators

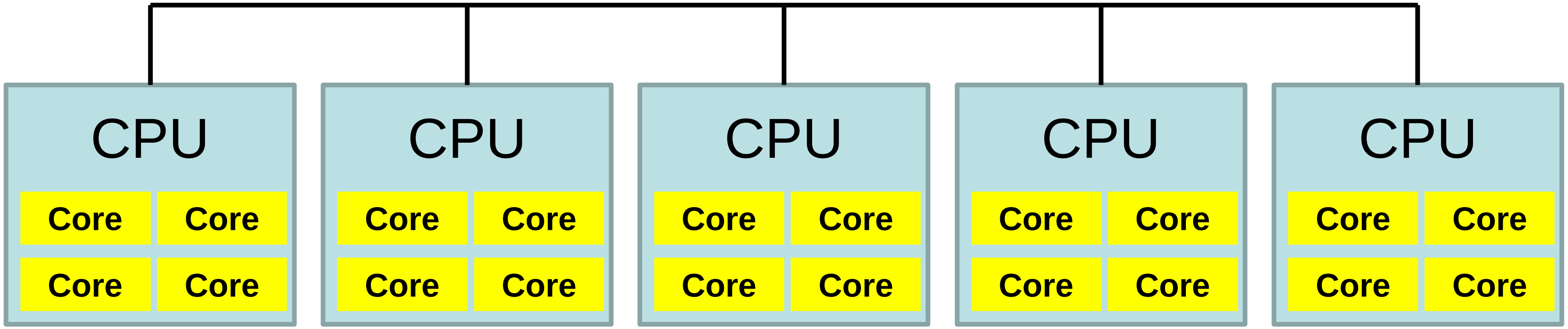
- GPU : Graphic Processing Unit
  - GPGPU: General Purpose GPU
  - O($10^{3+}$) cores
  - High Memory Bandwidth
  - (was) cheap
  - NO stand-alone operations
    - Host CPU needed
  - Programming: CUDA, OpenACC etc.
- NVIDIA, AMD, Intel …



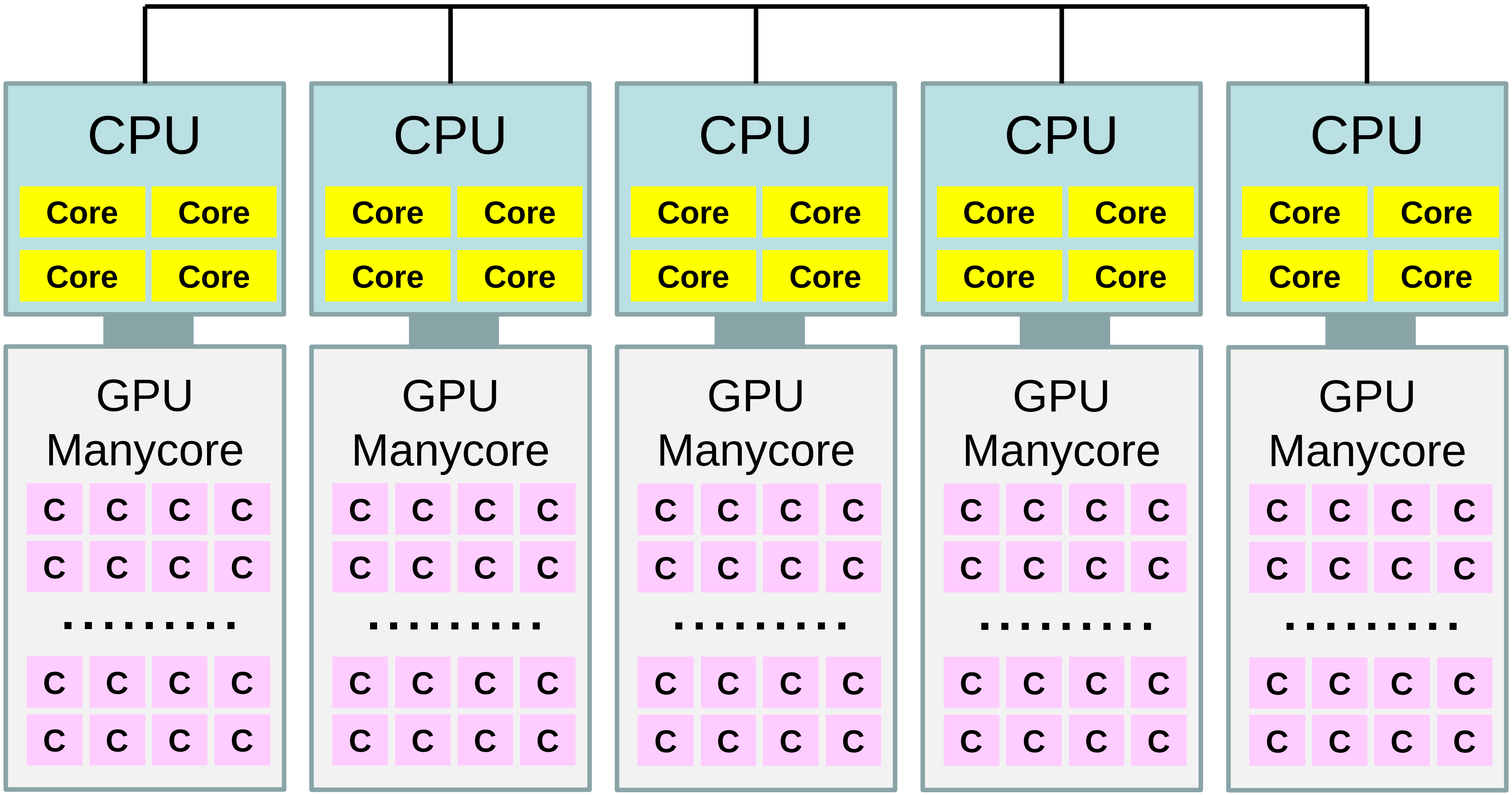



# Parallel Supercomputers

## Multicore CPU’s are connected through network



# Supercomputers with Heterogeneous/Hybrid Nodes



# Performance of Supercomputers

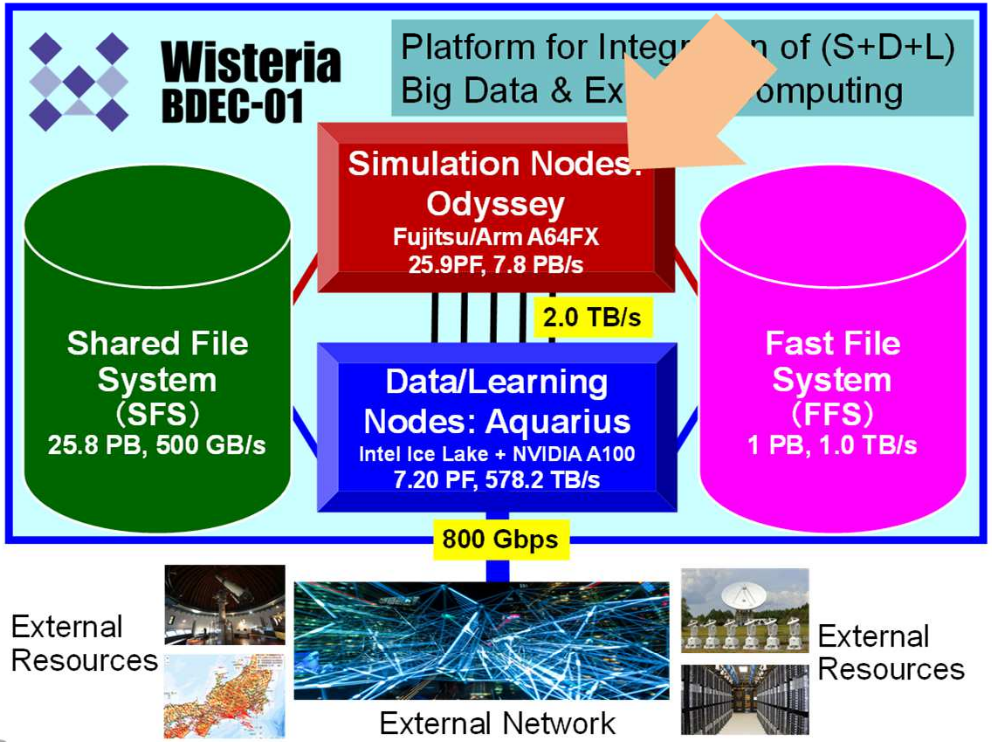
- Performance of CPU: Clock Rate
- FLOPS (Floating Point Operations per Second)
  - Real Number
- Recent Multicore CPU
  - 4-8 FLOPS per Clock
  - (e.g.) Peak performance of a core with 3GHz
    - $3 \times 10^9 \times 4$(or 8)=12(or 24)$\times 10^9$ FLOPS=12(or 24)GFLOPS

    - $10^6$ FLOPS= 1 Mega FLOPS = 1 MFLOPS
    - $10^9$ FLOPS= 1 Giga FLOPS = 1 GFLOPS
    - $10^{12}$ FLOPS= 1 Tera FLOPS = 1 TFLOPS
    - $10^{15}$ FLOPS= 1 Peta FLOPS = 1 PFLOPS
    - $10^{18}$ FLOPS= 1 Exa FLOPS = 1 EFLOPS

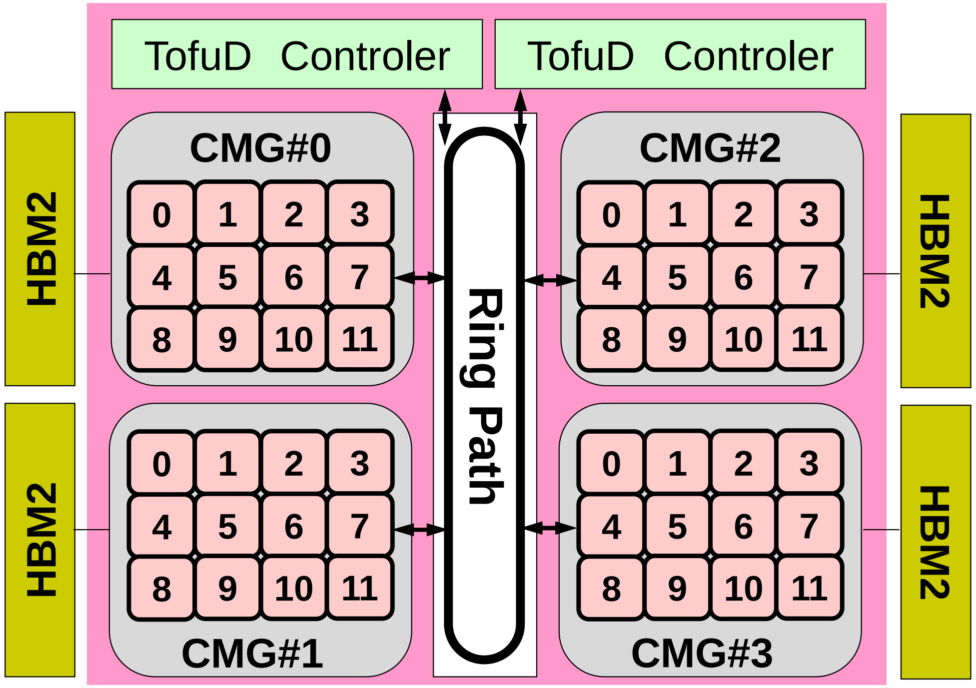
# Wisteria/BDEC-01 (Odyssey)

**The 1st BDEC System (Big Data & Extreme Computing)**
**Platform for Integration of (S+D+L)**

- Operation starts on May 14, 2021
- 33.1 PF, 8.38 PB/sec by **Fujitsu**
  - ~4.5 MVA with Cooling, ~360m²
- **2 Types of Node Groups**
  - Hierarchical, Hybrid, Heterogeneous (h3)
  - **Simulation Nodes: Odyssey**
    - **Fujitsu PRIMEHPC FX1000 (A64FX), 25.9 PF**
      - **7,680 nodes (368,640 cores), Tofu-D**
      - **General Purpose CPU + HBM**
      - **Commercial Version of "Fugaku"**
  - Data/Learning Nodes: Aquarius
    - Data Analytics & AI/Machine Learning
    - Intel Xeon Ice Lake + NVIDIA A100, 7.2PF
      - 45 nodes (90x Ice Lake, 360x A100), IB-HDR
    - Some of the DL nodes are connected to external resources directly
- File Systems: SFS (Shared/Large) + FFS (Fast/Small)



# A64FX Processor on Odyssey




| | |
|---|---|
| Name | A64FX |
| Processor # (Core #) | 1 (48+ 2or4 Assistant Cores) |
| Frequency | 2.2 GHz |
| Peak Performance | 3.3792 TFLOPS |
| Memory Size | 32 GiB |
| Memory Bandwidth | 1,024 GB/s |
| L1 Cache | 64 KiB/core (Inst/Data) |
| L2 Cache | 8 MiB/CMG |

- 4 CMG's (Core Memory Group), 12 cores/CMG
  - 48 Cores/Node (Processor)
  - 2.2GHz × 32DP × 48= 3379.2 GFLOPS= 3.3792 TFLOPS
- NUMA Architecture (Non-Uniform Memory Access)
  - Each core of a CMG can access to the memory on other CMG's
  - Utilization of the local memory is more efficient

# TOP 500 List

http://www.top500.org/

- Ranking list of supercomputers in the world
- Performance (FLOPS rate) is measured by "Linpack" which solves large-scale linear equations.
  - Since 1993
  - Updated twice a year (International Conferences in June and November)
- **"Fugaku" has been #1 from June 2020 to Nov.2021 (4 times)**
- **"Frontier" is the 1st Exaflop System in June 2022**
- Linpack
  - iPhone version is available

$10^{19}$= 10 ExaFlops

$10^{18}$= 1 ExaFlops

$10^{17}$=100 PetaFlops

$10^{16}$= 10 PetaFlops

$10^{15}$= 1 PetaFlops

$10^{14}$=100 TeraFlops

$10^{13}$= 10 TeraFlops

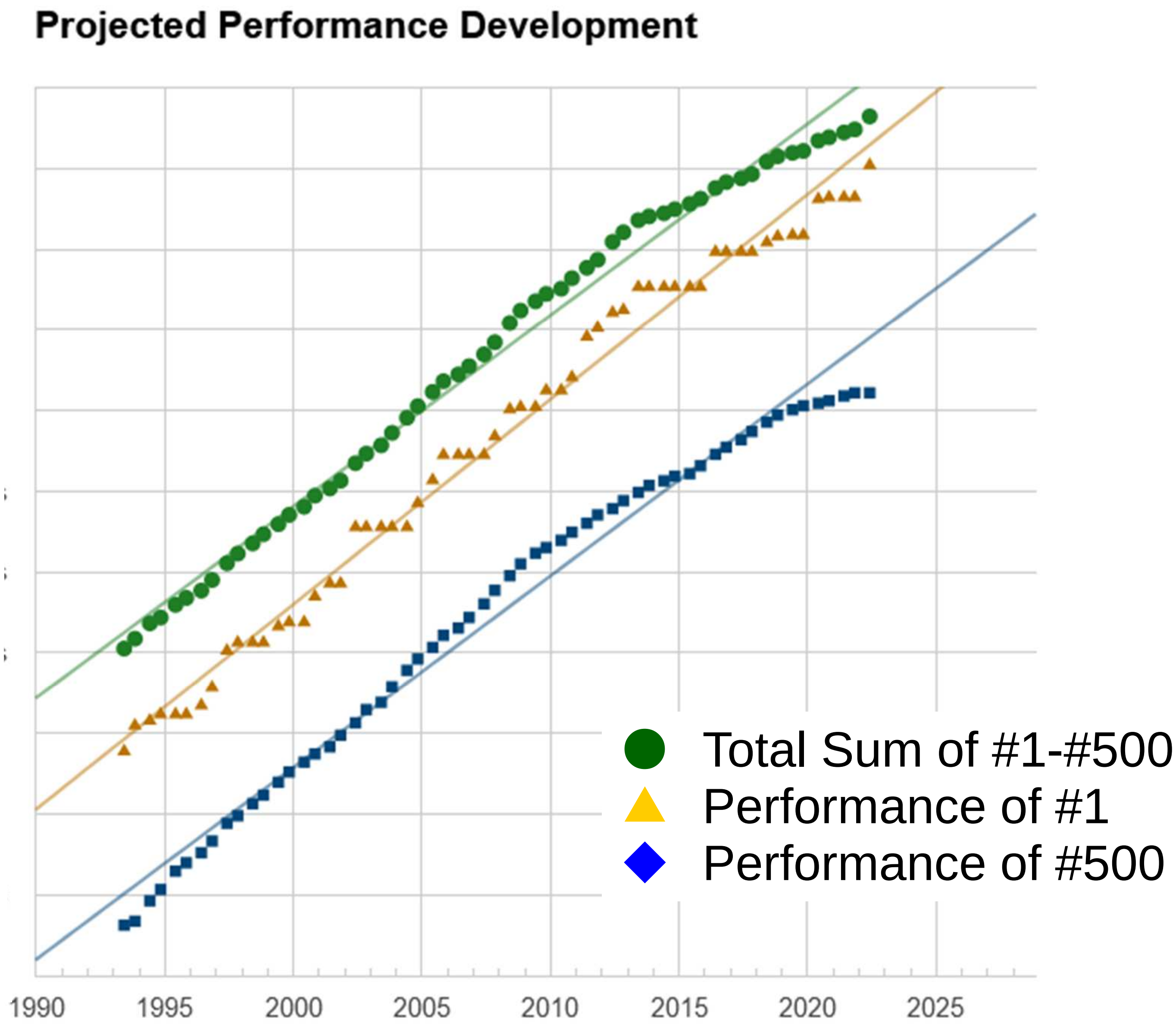
$10^{12}$= 1 TeraFlops

$10^{11}$=100 GigaFlops

$10^{10}$= 10 GigaFlops

$10^{9}$= 1 GigaFlops

$10^{8}$=100 MegaFlops




http://www.top500.org/

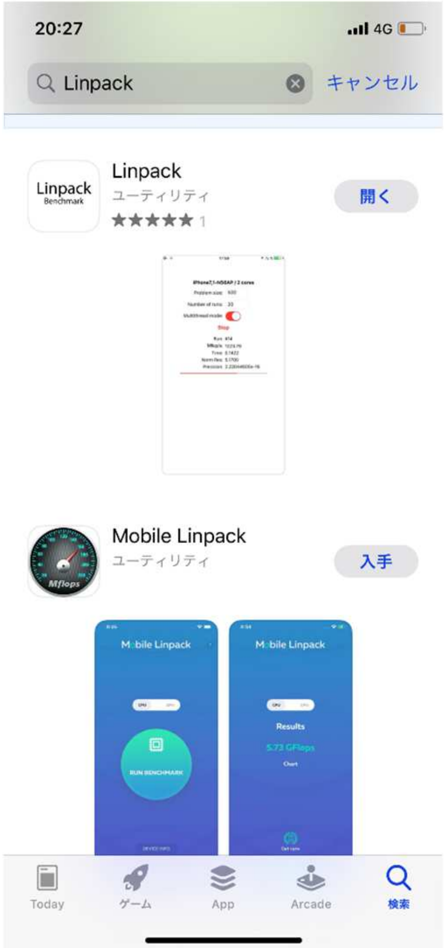
# 63rd TOP500 List (June, 2024)

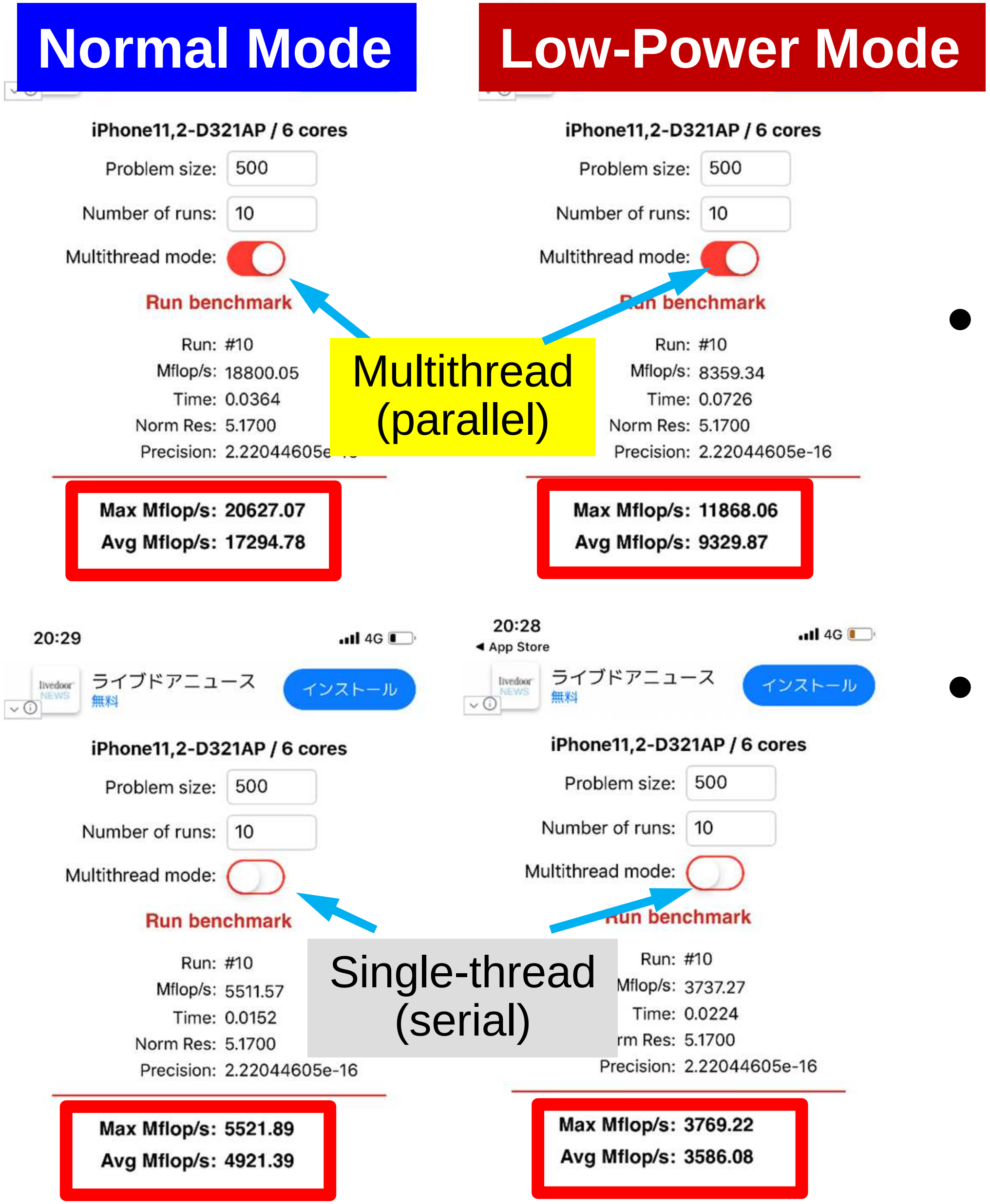
http://www.top500.org/

$R_{max}$: Performance of Linpack (TFLOPS)
$R_{peak}$: Peak Performance (TFLOPS),
Power: kW

| | Site | Computer/Year Vendor | Cores | $R_{max}$ (PFLOPS) | $R_{peak}$ (PFLOPS) | GFLOPS/W | Power (kW) |
|---|---|---|---|---|---|---|---|
| 1 | **Frontier, 2022, USA**<br>DOE/SC/Oak Ridge National Laboratory | HPE Cray EX235a, AMD Optimized 3rd Gen. EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11 | 8,699,904 | 1,194.00<br>(=1.194 EF) | 1,679.82<br>71.1 % | 52.93 | 22,703 |
| 2 | **Aurora, 2023, USA**<br>DOE/SC/Argonne National Laboratory | HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel | 9,264,128 | 1,012.00 | 1,980.01<br>51.1 % | 26.15 | 24,687 |
| 3 | **Eagle, 2023, USA**<br>Microsoft | Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR | 1,123,200 | 561.20 | 846.84<br>66.3 % | | |
| 4 | **Fugaku, 2020, Japan**<br>R-CCS, RIKEN | Fujitsu PRIMEHPC FX1000, Fujitsu A64FX 48C 2.2GHz, Tofu-D | 7,630,848 | 442.01 | 537.21<br>82.3 % | 14.78 | 29,899 |
| 5 | **LUMI, 2022, Finland**<br>EuroHPC/CSC | HPE Cray EX235a, AMD Optimized 3rd Gen. EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11 | 2,752,703 | 379.70 | 531.51<br>71.4 % | 53.43 | 7,107 |
| 6 | **Alps, 2024, Switzerland**<br>Swiss National Supercomputing Centre (CSCS) | HPE Cray EX254n, NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11 | 1,305,600 | 270.00 | 353.75<br>76.3 % | 51.98 | 7,107 |
| 7 | **Leonard, 2022, Italy**<br>EuroHPC/Cineca | BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64GB, Quad-rail NVIDIA HDR100 | 1,824,768 | 241.20 | 306.31<br>78.7 % | 32.19 | 7,494 |
| 8 | **MareNostrum 5 ACC, 2023, Spain**<br>EuroHPC/BSC | BullSequana XH3000, Xeon Platinum 8460Y+ 40C 2.3GHz, NVIDIA H100 64GB, Infiniband NDR200, EVIDEN | 663,040 | 175.30 | 249.44<br>70.3 % | 42.15 | 4,159 |
| 9 | **Summit, 2018, USA**<br>DOE/SC/Oak Ridge National Laboratory | IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR InfiniBand | 2,414,592 | 148.60 | 200.79<br>74.0 % | 14.72 | 10,096 |
| 10 | **Eos NVIDIA DGX SuperPOD**<br>NVIDIA Corporation | NVIDIA DGX H100, Xeon Platinum 8480C 56C 3.8GHz, NVIDIA H100, Infiniband NDR400, Nvidia | 485,888 | 121.40 | 188.65<br>64.4 % | | |
| 11 | **Venado, 2024, USA**<br>DOE/NNSA/LANL | HPE Cray EX254n, NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11 | 481,440 | 98.51 | 130.44<br>75.5 % | 59.29 | 1,662 |
| 17 | **CEA-HE, 2024, France**<br>CEA | BullSequana XH3000, Grace Hopper Superchip 72C 3GHz, NVIDIA GH200 Superchip, Quad-Rail BXI v2, EVIDEN | 389,232 | 57.11 | 112.56<br>50.7 % | 47.32 | 1,207 |
| 31 | **TSUBAME 4.0, 2024, Japan**<br>Tokyo Institute of Technology | HPE Cray XD665, AMD EPYC 9654 96C 2.4GHz, NVIDIA H100 SXM5 94 GB, Infiniband NDR200 | 172,800 | 25.46 | 59.40<br>42.9 % | 34.78 | 732 |
| 39 | **ABCI 2.0, 2021, Japan**<br>AIST | Fujitsu PRIMERGY GX2570 M6, Xeon Platinum 8360Y 36C 2.4GHz, NVIDIA A100 SXM4 40 GB, InfiniBand HDR | 504,000 | 22.21 | 54.34<br>40.9 % | 13.88 | 1,600 |
| 40 | **Wisteria/BDEC-01 (Odyssey), 2021, Japan**<br>ITC, University of Tokyo | Fujitsu PRIMEHPC FX1000, A64FX 48C 2.2GHz, Tofu interconnect D | 368,640 | 22.12 | 25.95<br>85.2 % | 15.07 | 1,468 |

# Linpack on My iPhone XS

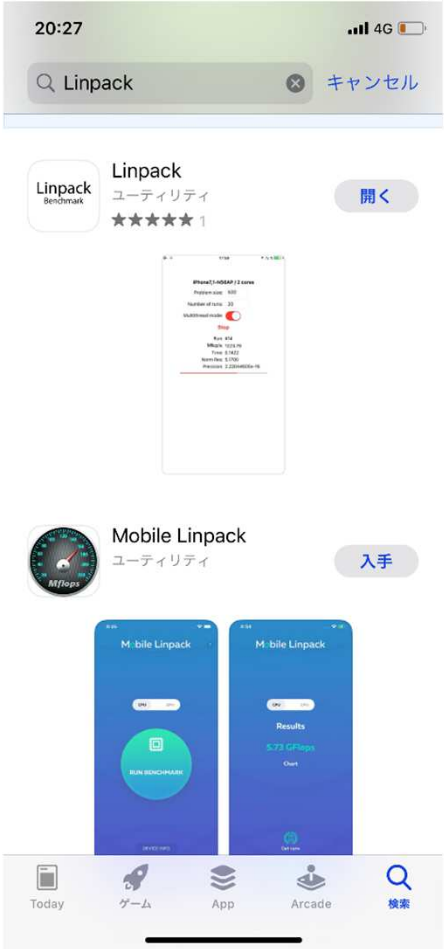




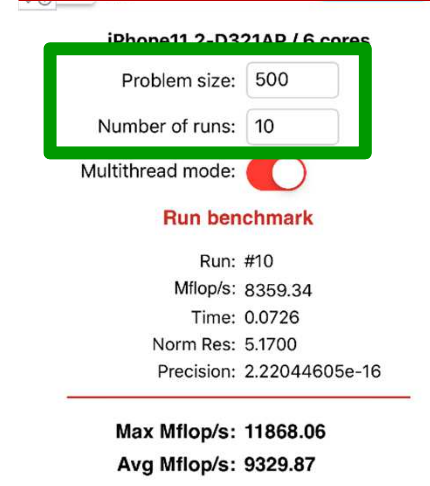
- Performance of my iPhone XS is about 20,000 Mflops
  - OBCX: 4.84 Tflops
- Cray-1S
  - Supercomputer of my company in 1985 with 80 Mflops
  - I do not know the price, but we had to pay 10 USD for 1 sec. computing !

# Linpack on My iPhone XS




**Normal Mode**


**Low-Power Mode**






- You can change Problem size, and # of runs.
  - "Size=500" means linear equations Ax=b with 500 unknowns are solved
- Actually, problem size affects performance of computing so much !!

# My New iPhone 15 Pro Max

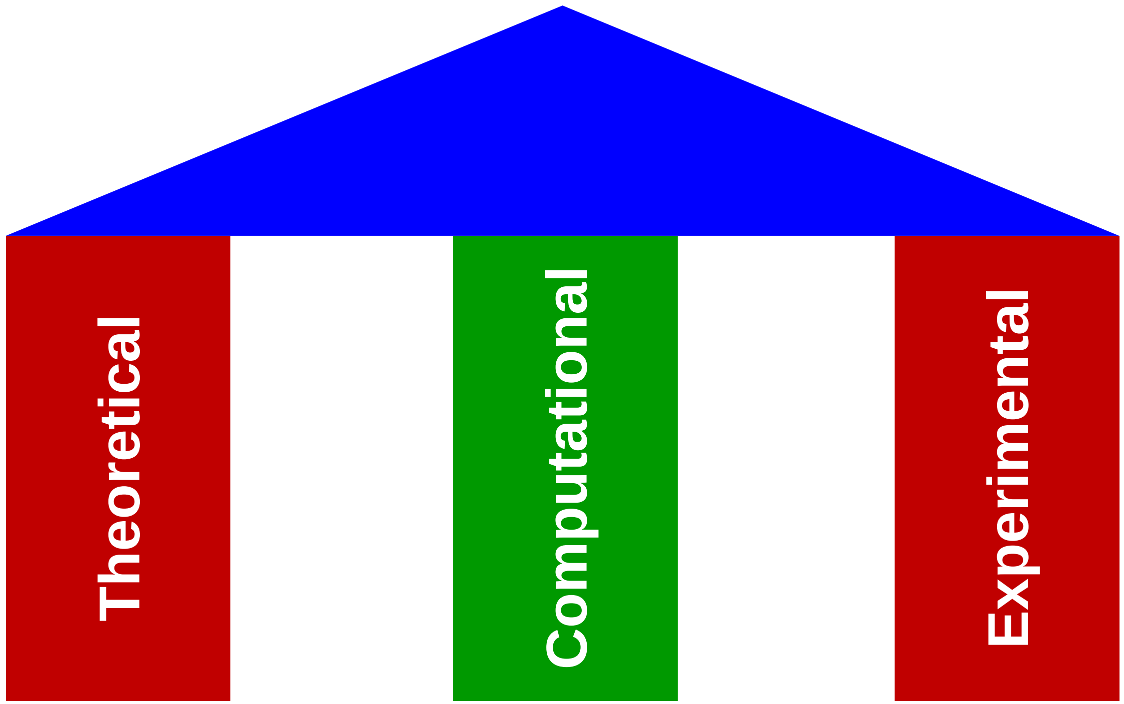
- Normal Mode, n=500, 10 times
- XS（A12, D321AP, 2.49GHz）
  - Single Core：    4.88 GF
  - Multicore (6):   13.15 GF
  - 177 g
- 15-PM（A17 Pro, D84AP, 3.78GHz）
  - Single Core：    8.27 GF
  - Multicore (6):   32.48 GF
  - 221 g
- Ratio
  - Frequency: 1.51
  - Single Core: 1.70, Multicore: 2.47

# Computational Science

## The 3rd Pillar of Science

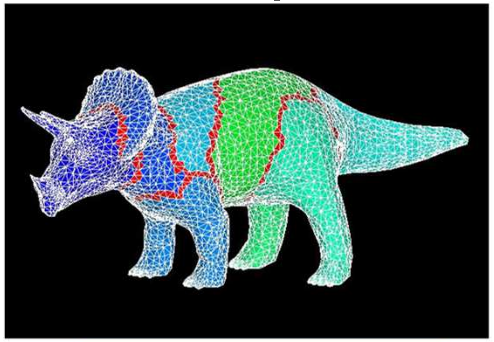
- Theoretical & Experimental Science
- Computational Science
  - The 3rd Pillar of Science
  - Simulations using Supercomputers

**CSE**

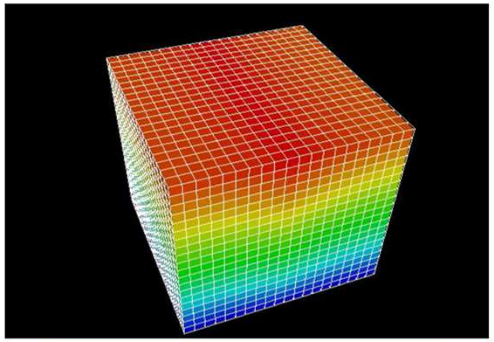


# Methods for Scientific Computing

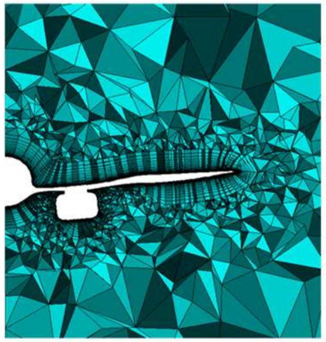
- Numerical solutions of PDE (Partial Diff. Equations)
- Grids, Meshes, Particles
  - Large-Scale Linear Equations
  - Finer meshes provide more accurate solutions



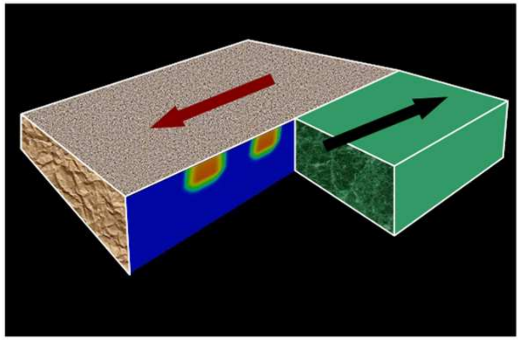
有限要素法
Finite Element Method
FEM



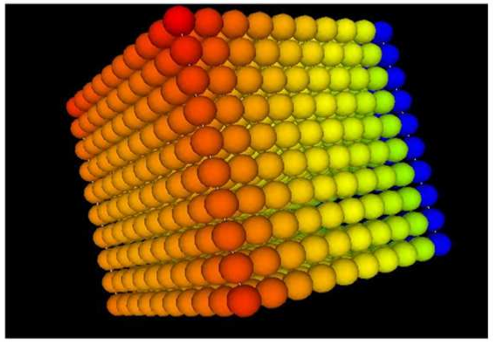
差分法
Finite Difference Method
FDM



有限体積法
Finite Volume Method
FVM



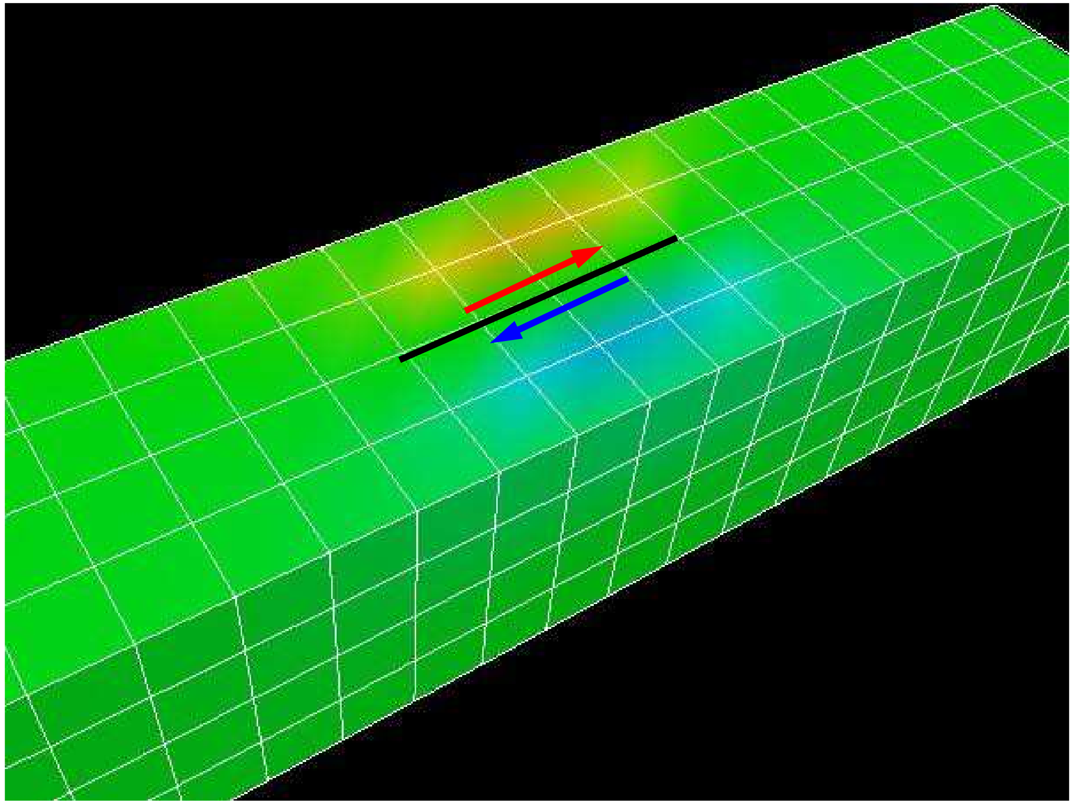
境界要素法
Boundary Element Method
BEM

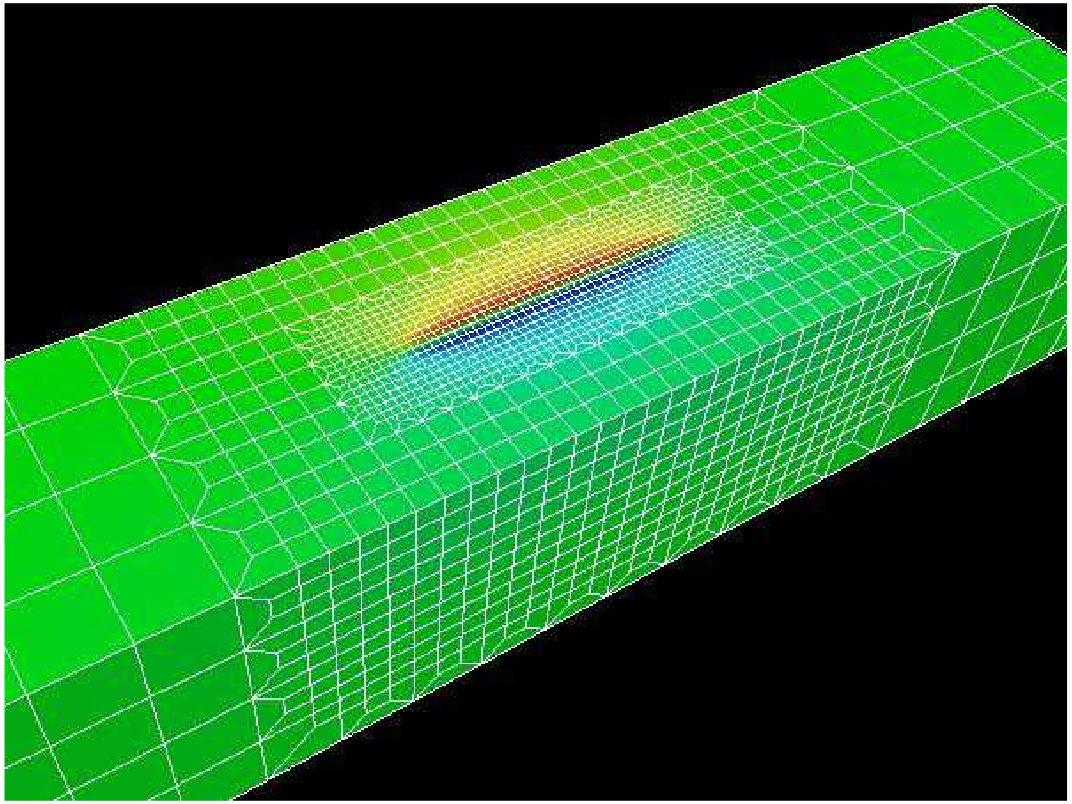


個別要素法
Discrete Element Method
DEM

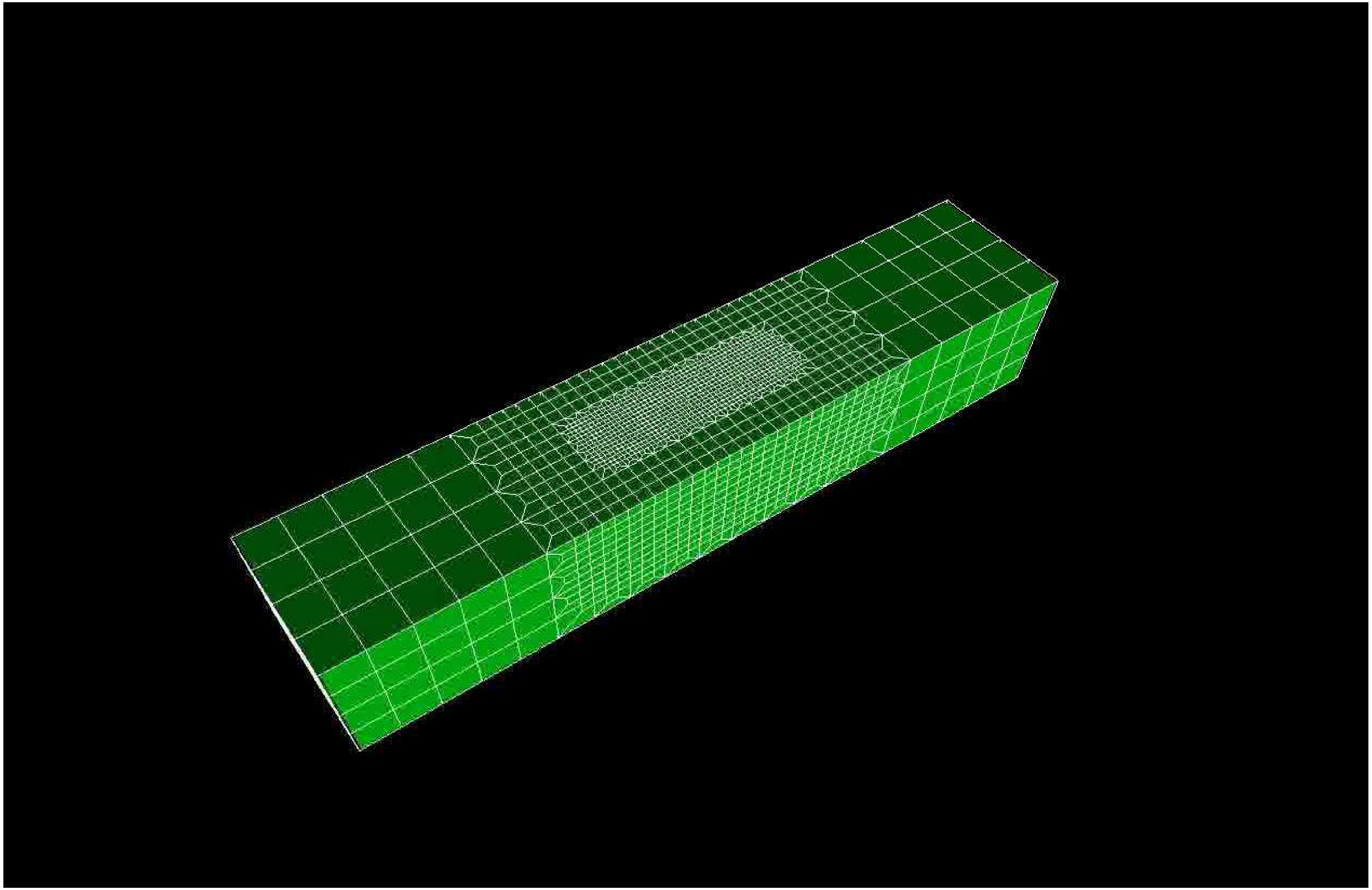
# 3D Simulations for Earthquake Generation Cycle San Andreas Faults, CA, USA

## Stress Accumulation at Transcurrent Plate Boundaries

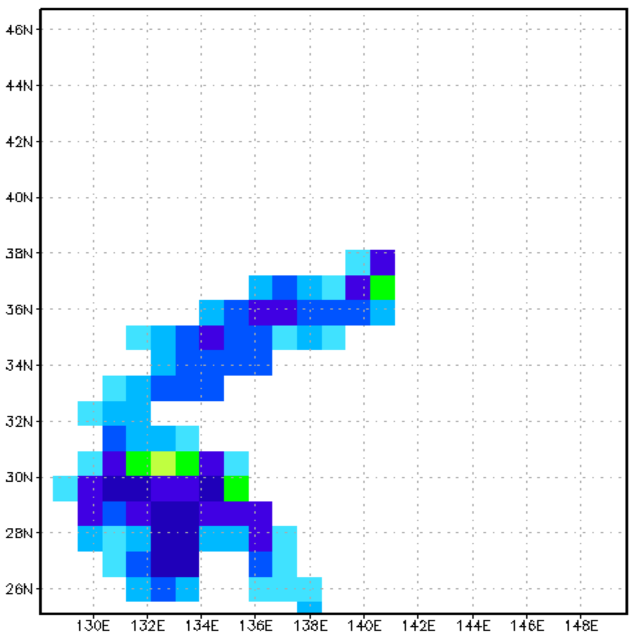




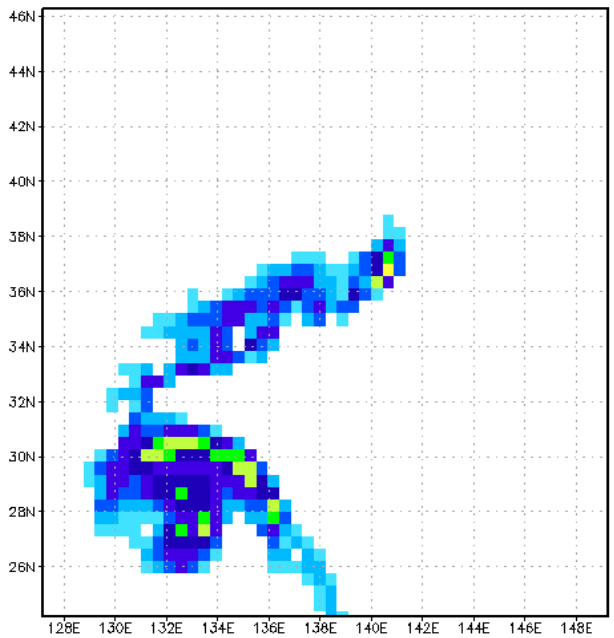
# Adaptive FEM: High-resolution needed at meshes with large deformation (large accumulation)



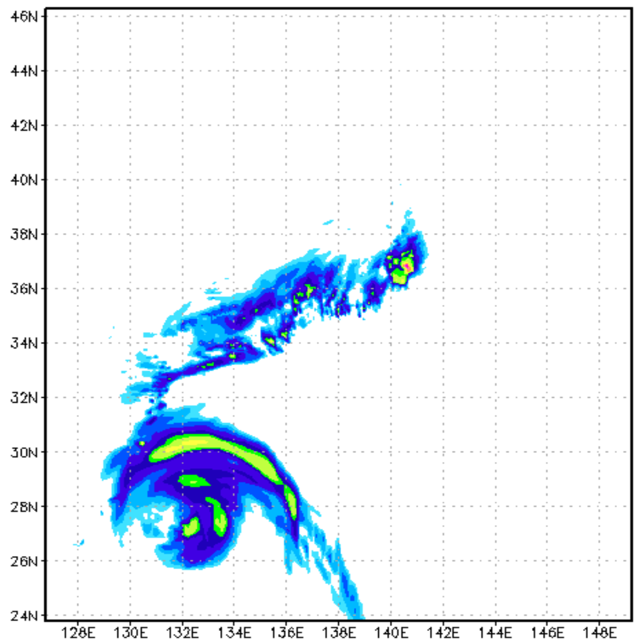
# Typhoon Simulations by FDM
# Effect of Resolution




**Δh=100km**



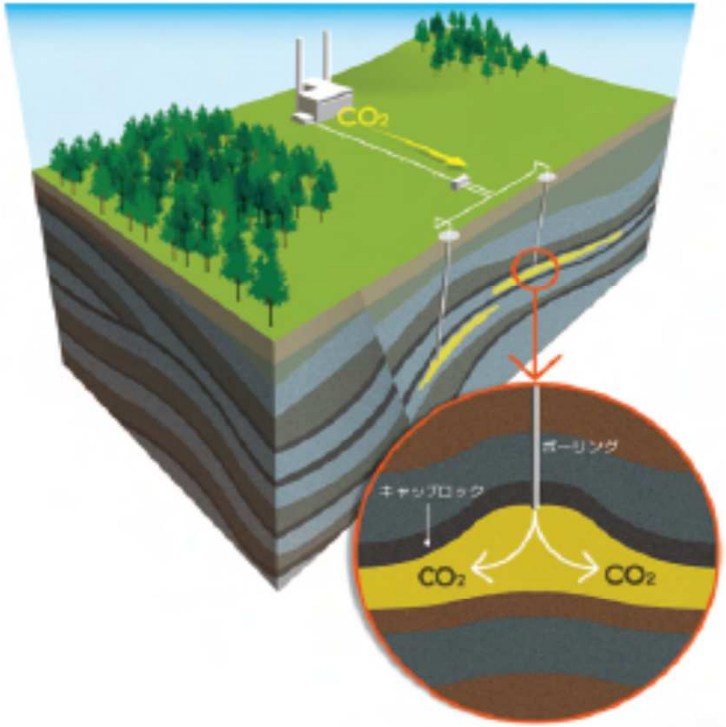

**Δh=50km**




**Δh=5km**

[JAMSTEC]

# Simulation of Geologic $CO_2$ Storage



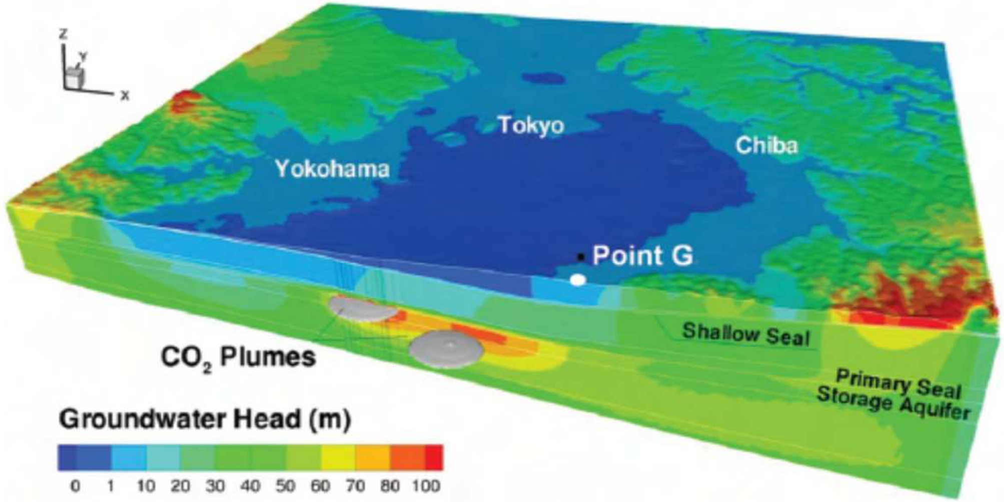
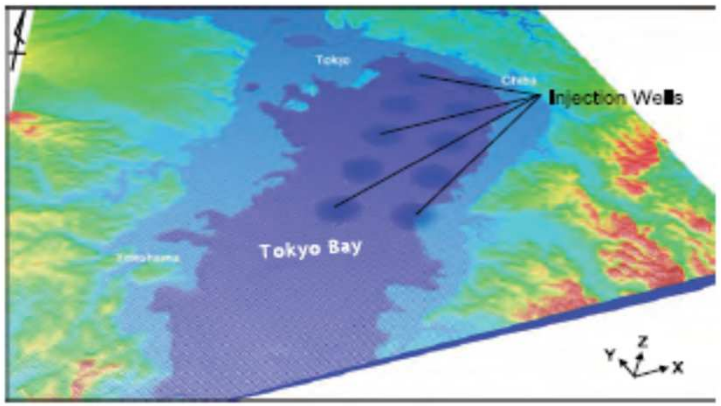
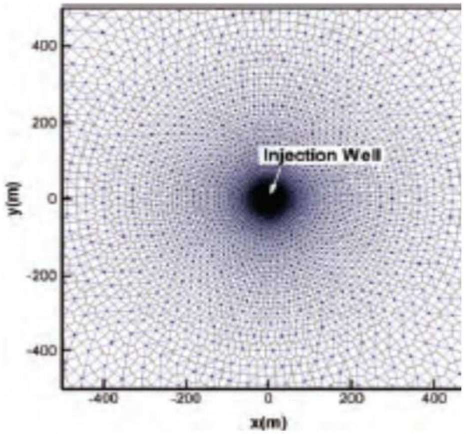
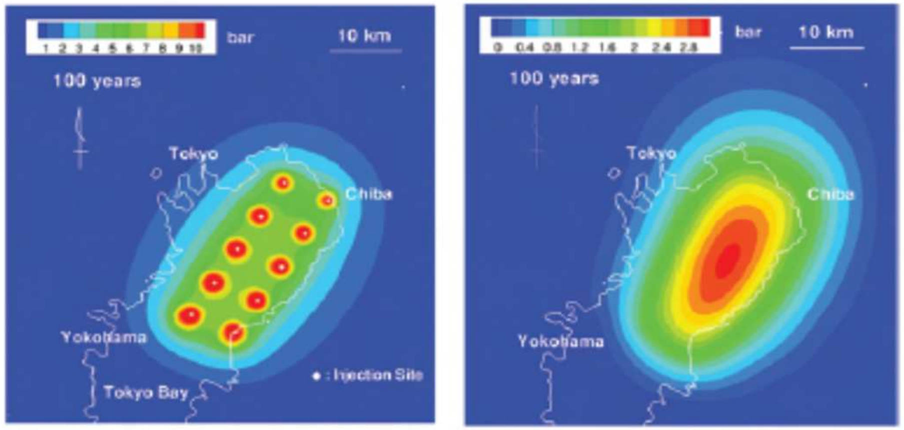




図-4　$CO_2$ 圧入後の地下水圧（全水頭換算）の分布（100 年後）




(a) 深部遮蔽層下面　　(b) 浅部遮蔽層下面

図-5　圧力上昇量の平面分布（初期状態からの増分、圧入開始から 100 年後

[Dr. Hajime Yamamoto, Taisei]

# Simulation of Geologic $CO_2$ Storage

- International/Interdisciplinary Collaborations
  - Taisei (Science, Modeling)
  - Lawrence Berkeley National Laboratory, USA (Modeling)
  - Information Technology Center, the University of Tokyo (Algorithm, Software)
  - JAMSTEC (Earth Simulator Center) (Software, Hardware)
  - NEC (Software, Hardware)
- 2010 Japan Geotechnical Society (JGS) Award

**Science**

**Modeling**

**Algorithm**

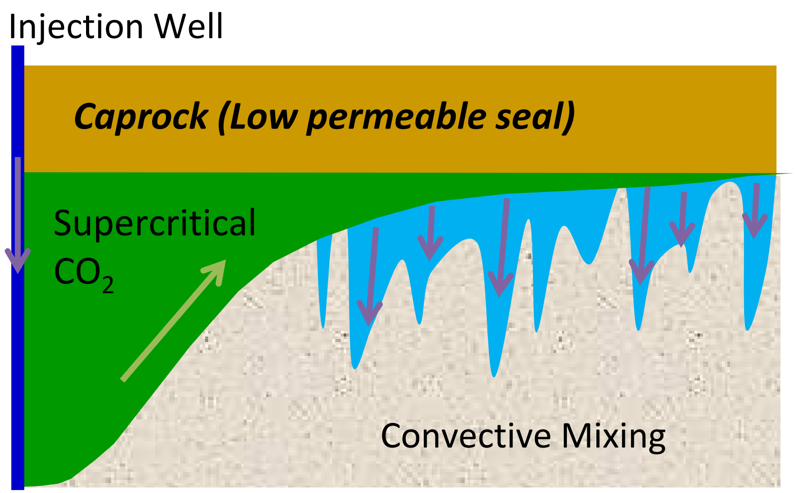
**Software**

**Hardware**

# Simulation of Geologic $CO_2$ Storage

- Science
  - Behavior of $CO_2$ in supercritical state at deep reservoir
- PDE's
  - 3D Multiphase Flow (Liquid/Gas) + 3D Mass Transfer
- Method for Computation
  - TOUGH2 code based on FVM, and developed by Lawrence Berkeley National Laboratory, USA
    - More than 90% of computation time is spent for solving large-scale linear equations with more than $10^7$ unknowns
- Numerical Algorithm
  - Fast algorithm for large-scale linear equations developed by Information Technology Center, the University of Tokyo
- Supercomputer
  - Earth Simulator II (NEX SX9, JAMSTEC, 130 TFLOPS)
  - Oakleaf-FX (Fujitsu PRIMEHP FX10, U.Tokyo, 1.13 PFLOPS

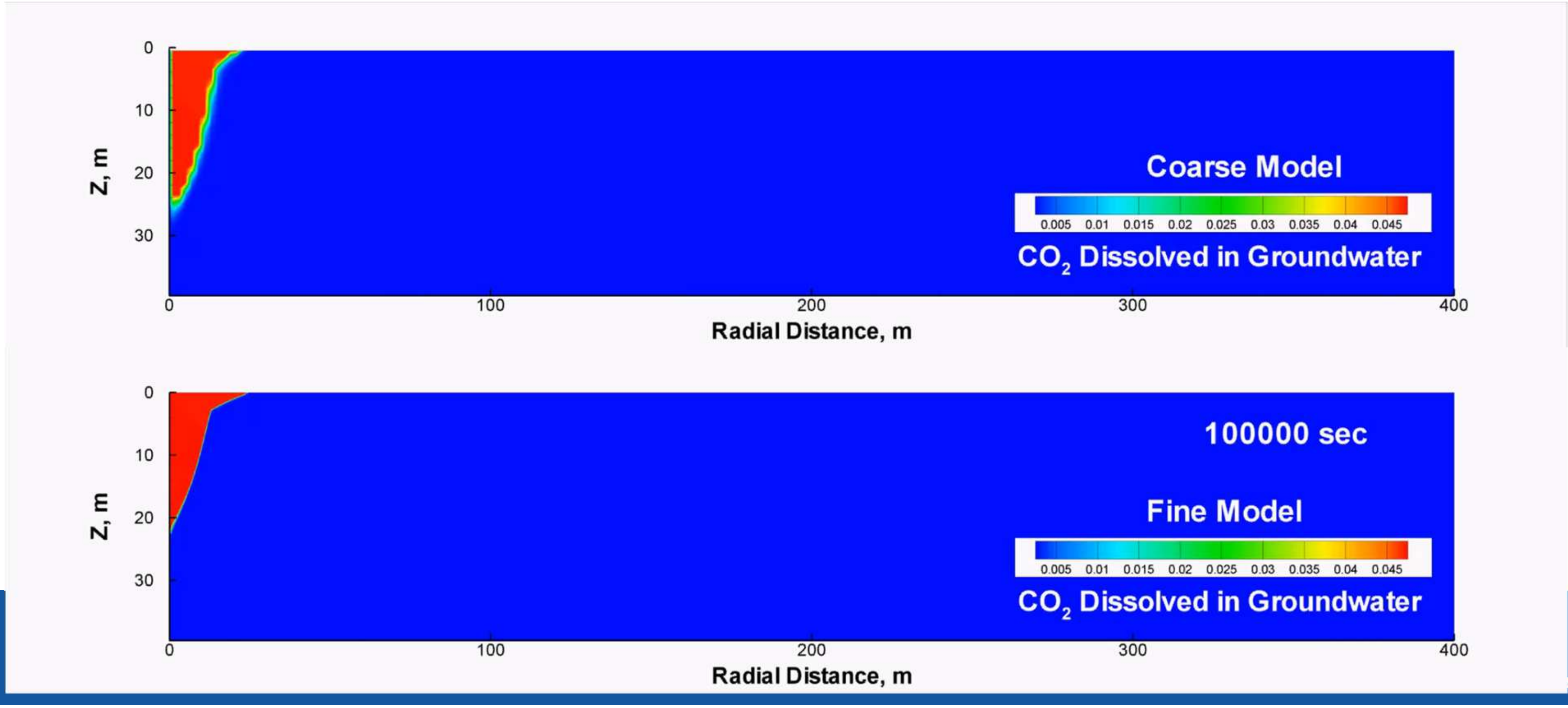
# Diffusion-Dissolution-Convection Process

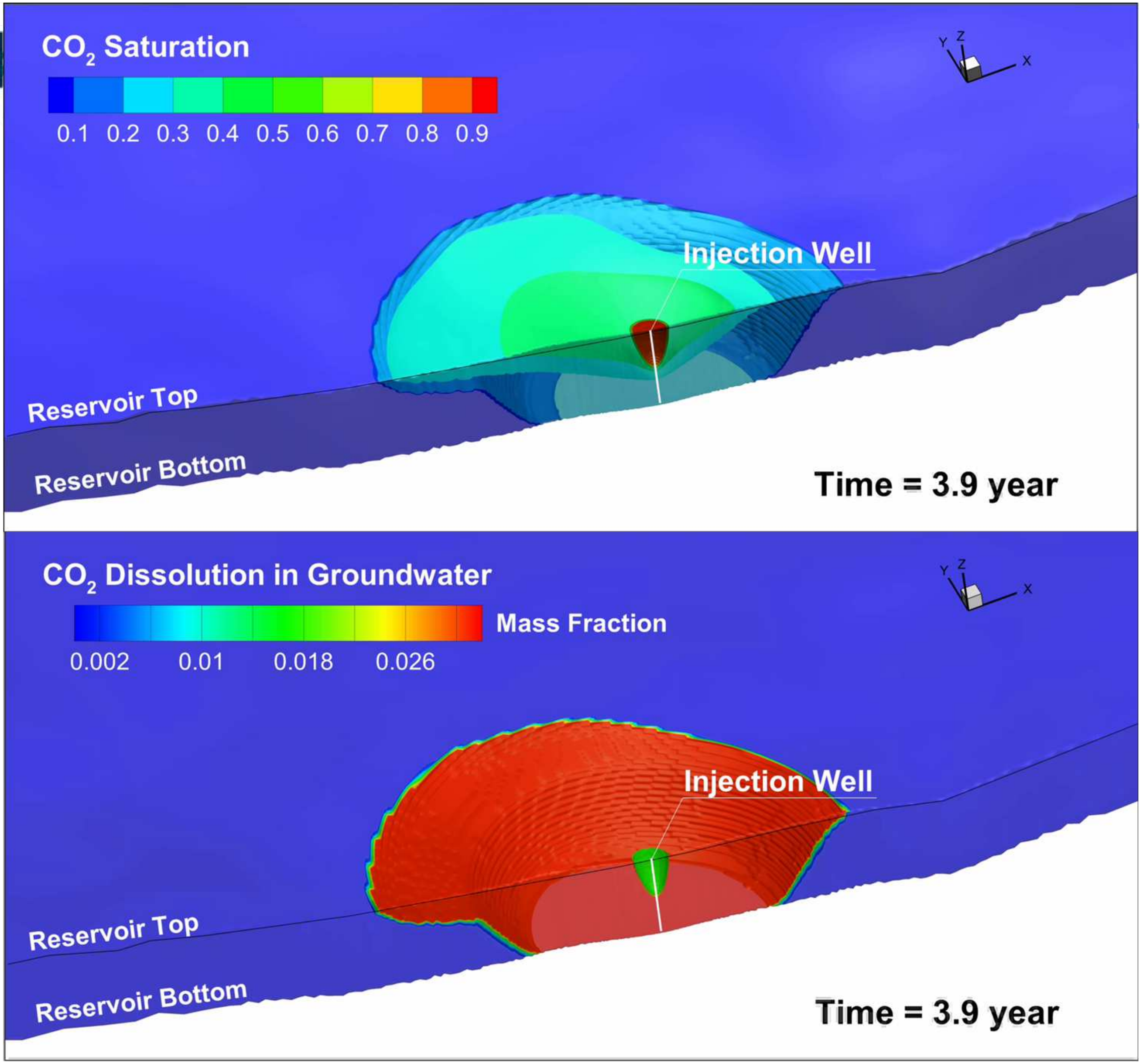



- **Buoyant sc$CO_2$ overrides onto groundwater**
- **Dissolution of $CO_2$ increases water density**
- **Denser fluid laid on lighter fluid**
- **Rayleigh-Taylor instability invokes convective mixing of groundwater**

**The mixing significantly enhances the $CO_2$ dissolution into groundwater, resulting in more stable storage**

**Preliminary 2D simulation (Yamamoto et al., GHGT11)** [Dr. Hajime Yamamoto, Taisei]






# Density convections for 1,000 years:

# Flow Model

Only the far side of the vertical cross section passing through the injection well is depicted.

Reservoir Condition

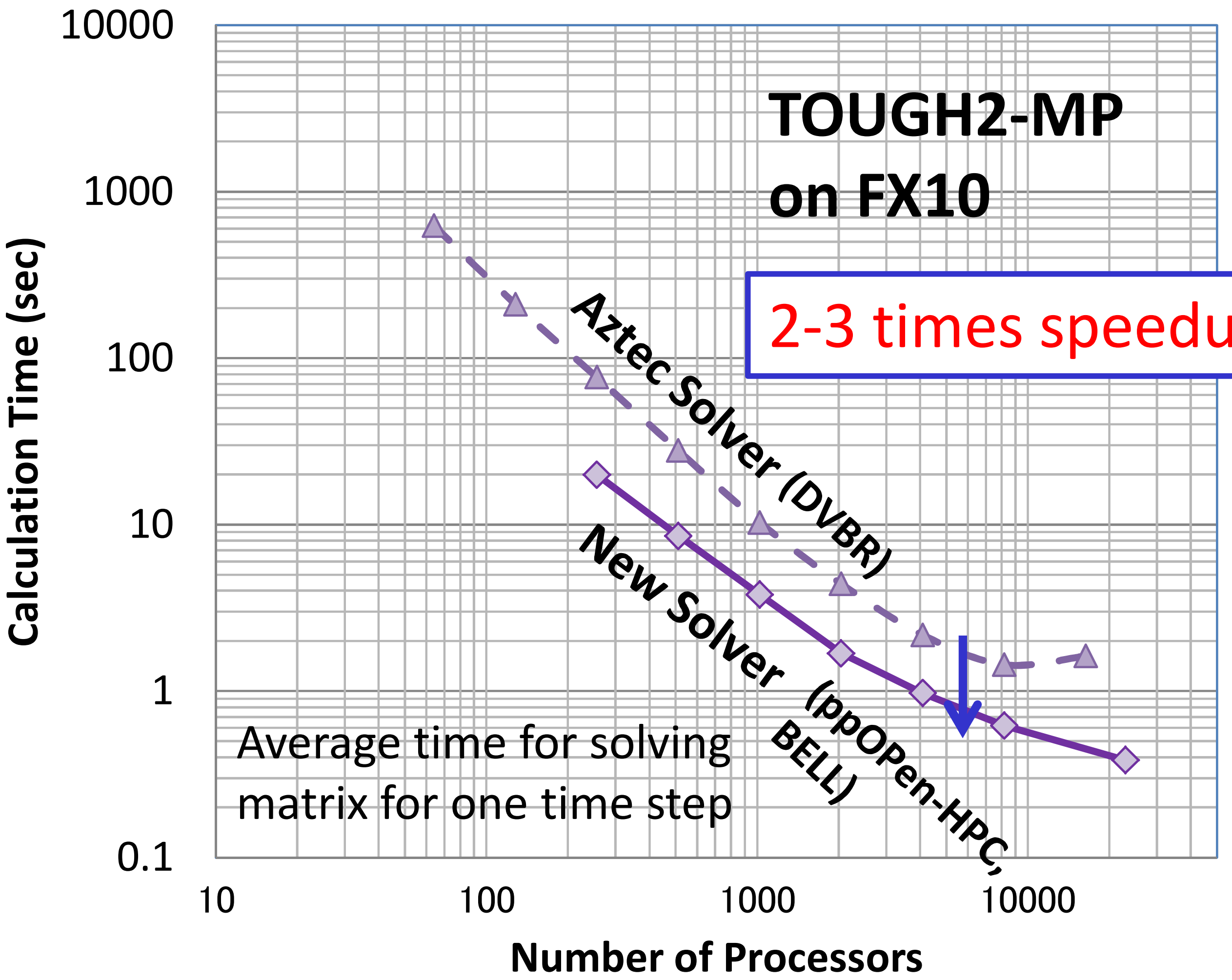
- Permeability: 100 md
- Porosity: 20%
- Pressure: 3MPa
- Temperature: 100$^{o}$C
- Salinity: 15wt%

[Dr. Hajime Yamamoto, Taisei]

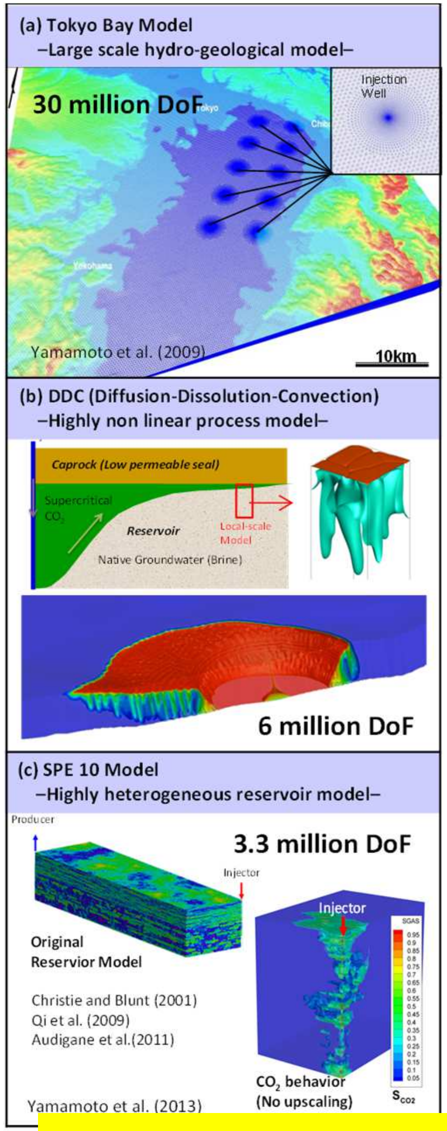
- The meter-scale fingers gradually developed to larger ones in the field-scale model
- Huge number of time steps (> $10^5$) were required to complete the 1,000-yrs simulation
- Onset time (10-20 yrs) is comparable to theoretical (linear stability analysis, 15.5yrs)

# Simulation of Geologic $CO_2$ Storage

30 million DoF (10 million grids × 3 DoF/grid node)




[Dr. Hajime Yamamoto, Taisei]




**Fujitsu FX10（Oakleaf-FX），30M DOF: 2x-3x improvement**

3D Multiphase Flow (Liquid/Gas) + 3D Mass Transfer

# Motivation for Parallel Computing, again

- Large-scale parallel computer enables fast computing in large-scale scientific simulations with detailed models. Computational science develops new frontiers of science and engineering.
- Why parallel computing ?
  - faster
  - larger
  - “larger” is more important from the view point of “new frontiers of science & engineering”, but “faster” is also important.
  - + more complicated
  - Ideal: Scalable
    - Weak Scaling, Strong Scaling

- Target: Parallel FEM
- Supercomputers and Computational Science
- **Overview of the Class**
- Future Issues

# Information

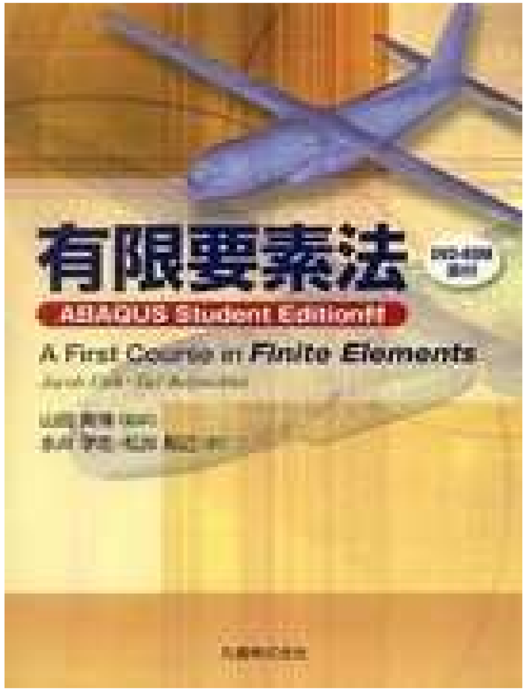
- Slack Channel
  - https://rikenihpcss2024.slack.com/
- Class Materials
  - http://nkl.cc.u-tokyo.ac.jp/2024-RIKEN-IHPCSS/

# Prerequisites

- Knowledge and experiences in fundamental methods for numerical analysis (e.g. Gaussian elimination, SOR)
- Knowledge and experiences in Unix/Linux
  - Essential for using Supercomputers
  - Google “Introduction to Unix”, “Introduction to Linux”
  - **`cd, ls, cp, mv, rm, scp`**
  - Editor: **`vi, emacs`** etc.
- Experiences in programming using FORTRAN or C

| September 2 (Mon) | | |
|---|---|---|
| 09:00-12:00 | Introduction, 1D FEM | Nakajima |
| 13:30-17:00 | 3D FEM | Nakajima |
| 17:00-18:00 | Fugaku Virtual Tour, Overview of Fugaku | Imamura |
| **September 3 (Tue)** | | |
| 09:00-12:00 | Road to Parallel FEM, Introduction to MPI (1/4) | Nakajima |
| 13:30-18:00 | Introduction to MPI (2/4) | Nakajima |
| **September 4 (Wed)** | | |
| 09:00-12:00 | Introduction to MPI (3/4) | Nakajima |
| 13:30-15:00 | Introduction to MPI (4/4) | Nakajima |
| 15:00-18:00 | Parallel FEM (1/2) | Nakajima |
| **September 9 (Mon)** | | |
| 09:00-12:00 | Parallel FEM (2/2) | Nakajima |
| 13:30-15:00 | Parallel Visualization | Nakajima |
| 15:00-18:00 | OpenMP/MPI Hybrid | Nakajima |
| **September 10 (Tue)** | | |
| 09:00-12:00 | PETSc | Imamura |
| 13:30-17:30 | CNN | Imamura |
| 17:30- | Closing | All |

# References




- Fish, Belytschko, A First Course in Finite Elements, Wiley, 2007
  - Japanese version is also available
  - “ABAQUS Student Edition” included
- Smith et al., Programming the Finite Element Method (4th edition), Wiley, 2004
  - Parallel FEM included
- Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Dover, 2000

- Target: Parallel FEM
- Supercomputers and Computational Science
- Overview of the Class
- **Future Issues**

# Technical Issues: Future of Supercomputers

- Power Consumption
  - 1MW=1,000kW~ 1M+ USD/yr, 200M+ JPY/yr
- Reliability, Fault Tolerance, Fault Resilience
- Scalability (Parallel Performance)

57

# Key-Issues towards Appl./Algorithms on Exa-Scale Systems

Jack Dongarra (ORNL/U. Tennessee) at ISC 2013

- Hybrid/Heterogeneous Architecture
  - Multicore + GPU/Manycores (Intel MIC/Xeon Phi)
    - Data Movement, Hierarchy of Memory
- Communication/Synchronization Reducing Algorithms
- Mixed Precision Computation
- Auto-Tuning/Self-Adapting
- Fault Resilient Algorithms
- Reproducibility of Results

# Supercomputers with Heterogeneous/Hybrid Nodes